

WOW!
EXCEL 4.0
(SEE PAGE 78)

WINDOWS MAGIC TRICKS • HARD DRIVE BASICS

COMPUTE

NOVEMBER 1992

KNOCKOUT NOTEBOOKS!

11 HOT NEW MACHINES DELIVER

FASTEST CPU_s

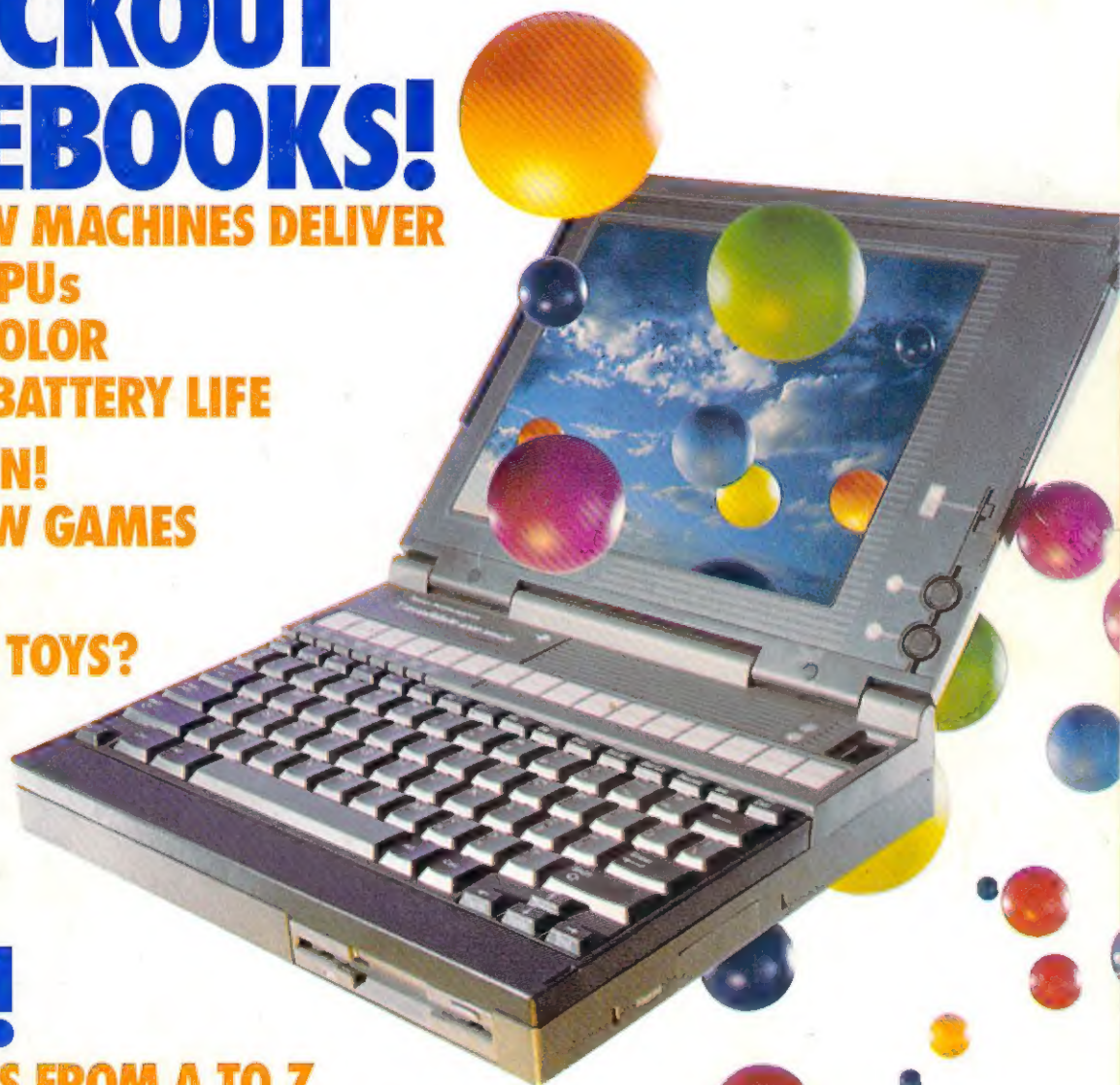
- **RICHEST COLOR**
- **LONGEST BATTERY LIFE**

SERIOUS FUN!

- **5 HOT NEW GAMES**

PALMTOPS

- **TOOLS OR TOYS?**



PLUS!

- **DATABASES FROM A TO Z**
- **CONTACT MANAGEMENT MADE EASY**
- **CORELDRAW! AND AMI PRO**

U.S. \$2.95





64/128 VIEW

If you've submitted a program recently and are still waiting to hear from us, please be patient.

Tom Netsel

There is good news and bad news to report this month. The good news is that we've been swamped with excellent type-in submissions. The bad news is that we've been swamped with excellent type-in submissions.

I made a pitch for programs in this column and in fillers elsewhere in the magazine encouraging submissions. Boy, did they work!

In fact, they've worked almost too well. For the past few months we've been deluged with good and great programs. We now have a large stack of them waiting to be reviewed. This has led to a new problem: Since it takes time to evaluate a program properly, we've been slow in mailing out contracts and rejection notices.

If you've submitted a program and haven't heard from us, please be patient. The quality as well as the quantity of submissions has been excellent this year, and we want to buy as many programs as we can. It just takes time to go through them all. When we've selected the programs that we plan to use in an issue and then come across another good program, we hate to reject it. We'll often hold it and use it the following month. But when we start holding too many programs, the system backs up. We'll get things moving shortly.

Actually, I love having too many submissions—so please keep them coming! With your help, we've been able to publish some great programs, and we want to continue the practice. A couple that come to mind from

last month are two SpeedScript spelling checkers for the 64 and 128.

I normally don't have two such similar programs in the same issue, but I thought that each spelling program would offer advantages to 64 and 128 users.

In this issue, we have a couple of programs that'll help programmers who work with sprites. These programs are geared more for the intermediate-to-advanced programmer who is already familiar with sprites and some of the problems associated with them.

MOB Master, by Hong Pham, adds ten new graphic commands to the 64 that make programming sprites much easier. Programming sprites on a 64 usually requires a lot of code filled with POKes, but MOB Master gives the 64 many of the same features and sprite commands found in BASIC 7.0 on the 128. With these commands, you'll find it much easier to define, position, and animate sprites.

Most people know that the 64 is capable of producing 16 different colors, but how would you like to boost that number to 136? You can with 136 Colors, a program by David Kwong.

Machine language programmers who use a 128 will want Bassem 128. Long a popular assembler for the 64, the 128 version is too large to type in, but it's available as this month's Gazette Disk bonus program.

I hope you'll find these and the other Gazette programs to be entertaining and informative. Be sure to let us know which programs you like or dislike. □

GAZETTE

64/128 VIEW

G-1

Great submissions flood Gazette office.
By Tom Netsel.

SID SIMPLIFIED

G-3

Cut through the confusion of programming the Sound Interface Device.
By Larry Cotton.

REVIEWS

G-10

Arachnophobia, Fun Graphics Machine, and Dweezilabel.

FEEDBACK

G-16

Questions and comments from our readers.

MACHINE LANGUAGE

G-18

Programming for speed, economy, or both?
By Jim Butterfield.

BEGINNER BASIC

G-20

Try a little machine language when your program needs a burst of speed.
By Larry Cotton.

D'IVERSIONS

G-21

Just call me Captain Future!
By Fred D'Ignazio.

GEOS

G-22

Assembling the ultimate GEOS system.
By Steve Vander Ark.

PROGRAMMER'S PAGE

G-24

Great tips from readers.
By Randy Thompson.

PROGRAMS

MOB Master

G-25

136 Colors

G-29

Tunnel Trap

G-33

BASIC Move and Save

G-36

Noah's Reader

G-38

Locate

G-38

Bug-Swatter

G-40

PUMP UP YOUR PRODUCTIVITY!

Harness the potential of your 64 and 128 with these powerful programs.

Get more work out of your 64 and 128 with these two new disk products from COMPUTE's Gazette – the 1992 Best of Gazette Utilities, and the Gazette Graphics Grab Bag!

The 1992 Best of Gazette Utilities

Seize control of your operating system and your world!

Here's what's on it—MetaBASIC 64, MetaBASIC 128, Quick, Sprint II, Ultrafont+, RAMDisk 64, RAMDisk 128, BASSEM, SciCalc 64, List Formatter, MegaSqueeze.

The Gazette Graphics Grab Bag

Do it all with Commodore graphics!

Here's what's on it—Starburst Graphics, Screen Designer 128, 128 Graphics Compactor, 64 Animator, VDC Graphics, Dissolve 128, Super Slideshow, 128 Animator, 1526 PrintScreen, Supratechnic, Medium-Resolution Graphics, Screen Maker, GAS!64—Special Edition, GAS!128—Special Edition.



**ORDER
THEM
TODAY!**

Extend Your Computer Power With This Powerful Software!

YES!

I want to pump up my productivity! Please send me the disks checked below at \$11.95 each.

☐ The 1992 Best of Gazette Utilities

☐ The Gazette Graphics Grab Bag

☐ Subtotal

☐ Sales Tax (Residents of NC and NY please add appropriate sales tax for your area. Canadian orders, add 7% goods and services tax.)

☐ Shipping and Handling (\$2.00 U.S. and Canada, \$3.00 surface mail, \$5.00 airmail per disk.)

☐ Total Enclosed

☐ Check or Money Order ☐ MasterCard ☐ VISA

Credit Card No. _____ Exp. Date _____

Signature _____
(Required)

Daytime Telephone No. _____

Name _____

Address _____

City _____

State/Province _____ ZIP/Postal Code _____

MasterCard and VISA accepted on orders with subtotal over \$20.

Mail this coupon to COMPUTE's 1991 Utilities, 324 West Wendover Ave., Ste. 200, Greensboro, NC 27408.



THE  SOUND
INTERFACE
DEVICE,  AN
INTEGRATED
 CIRCUIT 
CHIP FONDLY
KNOWN AS SID,

RESIDES DEEP IN THE ELECTRONIC
INNARDS OF THE  **SID SIMPLIFIED** 
COMMODORE 64 AND 128 COMPUTERS.
IT HAS THE ABILITY TO LET YOUR
COMPUTER  PLAY, SING, MOAN,
TALK, RING, THUMP, SCREAM, AND
WHISPER. THIS CHIP ALONE HAS BEEN
AT  **BY LARRY COTTON**  LEAST
PARTIALLY RESPONSIBLE FOR THE
FACT THAT  COMMODORE STILL
BUILDS THE 64 ALMOST NINE YEARS
AFTER ITS SPLASHY INTRODUCTION—
A COMPUTER LONGEVITY RECORD. 

With all SID's capabilities, programming it in BASIC 2.0 remains an exercise in tedium, because of the many POKEs required to access the chip. (BASIC's POKE puts a number from 0 to 255 into a specific location in the computer.)

Fortunately for 128 owners, Commodore included with that machine a much-advanced BASIC 7.0, which does support SID and makes programming sounds much easier.

This article will attempt to cut through the confusion of programming SID and show you, step by step, how to access this marvelous chip. I'll confine my remarks to BASIC 2.0's commands, common to both the 64 and 128, and I'll show you how to cut down drastically on the number of POKEs. We'll start with the very simplest exercises and progress to the more advanced. If you'll stay with me from the beginning, you'll be pleased with the results.

If you're confused about programming SID, it will first be necessary to power down your own mind to rid it of all past frustrating programming sessions. Start from scratch. Remember that we're talking about only 29 of the 64's 64,000 or so memory registers. How complicated can they be?

Voices

A human being has only 1 voice; a saxophone has only 1 voice. A six-string guitar has 6; a piano, 88. SID has 3. Think of SID as a three-string guitar. That is, up to three notes can be played simultaneously, each under separate control (except for volume).

We'll limit our initial discussions to voice 1, which occupies SID's first seven memory registers. Remember that number, 7; it'll crop up again.

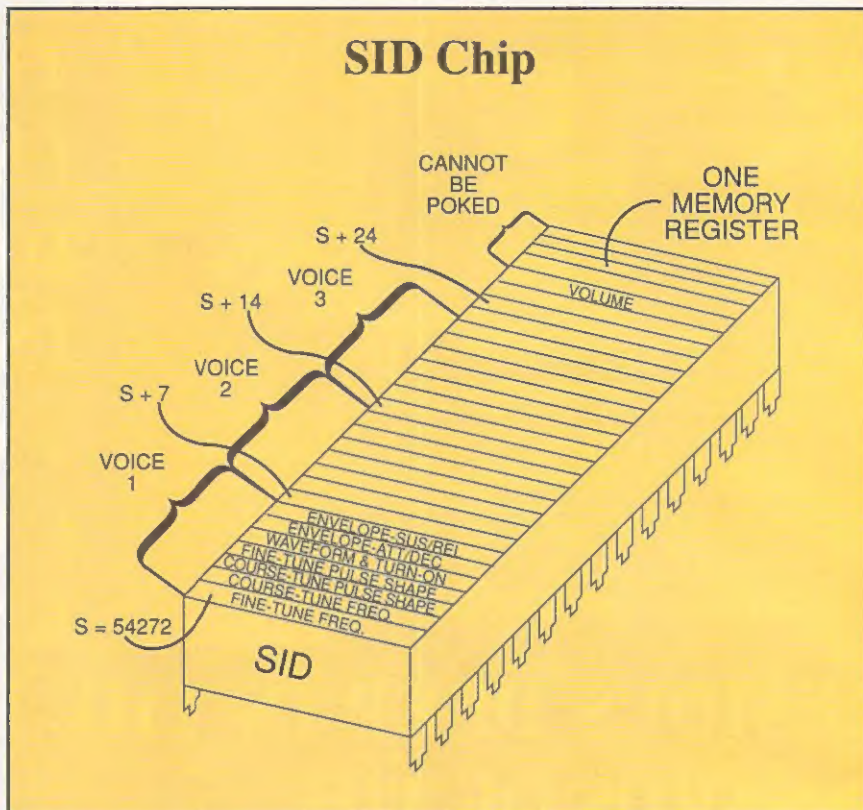
Order of POKEs

Here's a subject rarely addressed and, I think, fairly critical to the success of SID programming: the order that the memory registers are poked. Here is the normal order for playing a simple sound.

1. Clear the chip.
2. Turn up the volume.
3. Wait.
4. Set a frequency.
5. Set an envelope.
6. Turn on a waveform.

Clear the Chip

SID occupies memory registers 54272 through 54300. All those registers (except the last four, which cannot be poked) should always be cleared of



their contents near the beginning of every BASIC program which uses sound. Here's how.

**10 S=54272: FOR J=S TO S+24: POKE J,0:
NEXT: REM CLEAR SID**

SID's first memory register should be defined as a constant; we'll use S. Then every other register may be defined as an offset of S. A FOR-NEXT loop pokes a 0 into each of the SID memory registers, effectively silencing the chip and preparing it for action.

Turn Up the Volume

SID's last pokable register is the volume control. Its range varies from 0 to 15, with 0 being the quietest setting. Let's turn the volume wide open with the following statement.

20 POKES+24,15: REM FULL VOLUME

Any memory register will accept values from 0 to 255, but 54296 uses only values from 0 (silent) to 15 (loud) to control volume. Normally, S+24 can keep a value of 15 throughout a BASIC program.

Wait

Turning up SID's volume makes a popping noise in the TV or monitor's speaker, and this can interfere with your carefully crafted sound. Always introduce a period of silence after first

turning up SID's volume. We'll show a do-nothing time delay, but ordinarily at this point in a program you'd be preparing the screen, reading data, setting variables, and so forth.

30 FOR T=1 TO 1000: NEXT

Set a Frequency

SID needs several other values poked to it before it will speak up. For instance, it needs a frequency. A frequency controls a note's pitch.

40 POKES+1,16: REM FREQUENCY

SID's voice 1 memory location 54273 (S+1) can use all values from 0 to 255. A value of, say, 5 produces sounds of low pitch (like a tuba). A value of 200 produces a high-pitched sound (like a piccolo).

Set an Envelope

What's an envelope? Nothing more complicated than how the volume of a single particular note (or sound effect) changes as it plays.

Think about the way a single guitar string sounds as it's being plucked. The guitar makes no sound at first, but its sound level rises from silence to maximum volume immediately after the string is plucked. That's called attack. The sound then gradually fades away to silence. This is called decay.

TENEX Introduces: WORLD'S LOWEST PRICE FOR AMIGA 500!

Amiga 500 Computer Basic Package

\$299⁹⁵



Commodore®
AMIGA®

**Free
Lemmings!**
with Amiga Upgrade Packages

Plus 3 Great Value Packages!

TENEX Bonus Pack

- Amiga 500 Basic Package
- Software Bundle #1
- Free Lemmings!
- TV Adapter

Total at Reg. Price \$364.85
Low TENEX Package Price **\$339.95**
You Save **\$24.90 More!**

TENEX Power Pack

- Amiga 500 Basic Package
- Software Bundle #1
- Software Bundle #2 & Joystick
- Free Lemmings!
- TV Adapter
- TENEX 500 Memory Expansion

Total at Reg. Price \$454.75
Low TENEX Package Price **\$399.95**
You Save **\$70.75 More!**

TENEX Super Graphics Pack

- Amiga 500 Basic Package
- Amiga 1084S Stereo Monitor
- Software Bundle #1
- Software Bundle #2 & Joystick
- Free Lemmings!
- TENEX 500 Memory Expansion

Total at Reg. Price \$739.75
Low TENEX Package Price **\$669.00**
You Save **\$54.80 More!**

Software Bundle #1 includes: Tetris, Where in the World is Carmen Sandiego, and Textcraft.
Software Bundle #2 includes: MasterType, F-40 Pursuit Simulator, Who! What! When! Where!, and Hole-in-One Miniature Golf.

Hot Printer Values!

\$179⁹⁵

\$136⁹⁹

STAR NX-1020 RAINBOW

Enjoy vibrant color output, with easy handling! Choose from three print modes: high-speed draft at 225cps, draft at 180cps, and near-letter-quality at 75cps. Features include a big 16KByte buffer, six resident fonts, 15 convenient front panel controls, the ability to print on four-part forms, a slide panel interface for interference-free paper movement, and more! Download as many as 255 characters for creating unique logos and fonts. You get Epson FX and IBM Proprinter III emulation. Two-year warranty. Sug. Retail \$329.00

NX-1020 Rainbow A51027 **\$179.95**

STAR NX-1001 Multi-Font

This 9-wire, dot matrix model provides versatility at a great price. Quietly print in near-letter-quality at 75cps. Choose top or rear paper paths and five resident fonts: Draft, Courier, Sans Serif, and Orator 1 & 2. Download 192 characters for designing custom fonts and logos. Your versatility is extended further with the ability to clear the 4KByte buffer with the push of a button. Two-year warranty. Sug. Retail \$249.00

NX-1001 Multi-Font 90895 **\$136.95**

OTHER PRINTERS

Star NX-2420 Rainbow	A51047	\$284.95
Star LaserPrinter 4	A57934	\$799.00
Star NX-2430 Multi-font	98734	\$229.95
Panasonic KX-P2180	A68846	\$179.95
Panasonic KX-P1180i	A68584	\$159.95
Panasonic KX-P2123	A72449	\$249.95
Panasonic KX-P1124i	A57112	\$289.95
Panasonic KX-P1123	A57108	\$199.95
Panasonic KX-P4410 Laser	A71385	\$649.95

CARDPRINT G-WIZ INTERFACE.

Connect any printer to your C64/128. Dumps high-res screens up to 18 times faster than competitive interfaces without buffers. 90-day warranty. From Supra. Sug. Retail \$69.95

G-Wiz Interface 34484 **\$39.95**

Commodore 64 Computer

Only

\$149⁹⁵

Factory
New!



Commodore 1541 II Disk Drive

Only

\$169⁹⁵

Factory
New!



Don't miss out on the lowest prices on the Amiga 500, plus a full line of Commodore and Amiga hardware, software, and accessories. Call today to receive your **FREE** catalog with the greatest prices on the most popular hardware and software!



IMPORTANT NOTE
All of our products are brand new and factory fresh! Don't be fooled by reconditioned products or off-brand substitutes. Trust TENEX to bring you the best quality at outstanding prices!



Shipping, Handling, Insurance

Order Amount	Charge
less than \$19.99	\$4.95
\$20.00-\$39.99	\$5.95
\$40.00-\$74.99	\$6.95
\$75.00-\$99.99	\$7.95
\$100.00-\$149.99	\$9.95
\$150.00-\$299.99	\$10.95
\$300.00-\$499.99	\$12.95
\$500.00-\$699.99	\$19.95
\$700.00-\$999.99	\$27.95
\$1000 & Over	2.8% of Order

TENEX

Computer Express

Order Today! Call 1-800-PROMPT-1

(1-800-776-6781)

56800 Magnetic Drive
Mishawaka, IN 46545
(219)259-7051 FAX (219)259-0300
We gladly accept mail orders!
Circle Reader Service Number 170

SID can create sounds quickly, like a guitar, or slowly, more like a bowed violin. It can also do two more things to a sound which a guitar can't. It can prolong a sound's volume at a particular level. This is called sustain. SID can also cause the sound to stop at a controllable rate with a process called release.

So, there you have it. The sound's envelope is made of attack, decay, sustain, and release. Each of these properties is controllable. For now though, the properties we'll use are attack and decay. A value of 12, in fact, poked to the envelope simulates the plucking of a guitar string. Later, we'll see how to determine values to poke. Where do we poke that envelope value? We poked the frequency into S+1, so the envelope must be poked into S+2, right? I'm afraid not; S+2 and S+3 are reserved for fine-tuning the pulse wave. S+4? Nope. That turns on voice 1. S+5 (54277) is voice 1's main envelope-controlling register.

50 POKES+5,12: REM ATTACK/DECAY

If you want to experiment with sustain and release, add this line.

52 POKES+6,4: REM SUSTAIN/RELEASE

SID AND VARIABLES

Using a variable such as F, instead of a number like 16, yields a whole new world of sounds. Here's an example which emulates a warning siren.

```
10 S=54272: FORJ=STOS+24: POKEJ,0:
NEXT: REM CLEAR SID
20 F=16: REM DEFINE VARIABLE
30 POKES+24,15: REM FULL VOLUME
40 FORT=1TO200: NEXT: REM SHORT
PAUSE
50 POKES+1,F: POKES+8,F*1.3: REM
COARSE FREQUENCIES
60 POKES,195: POKES+7,31: REM FINE-
TUNE FREQUENCIES
70 POKES+5,12: POKES+12,12: REM
ATTACK/DECAY
80 POKES+6,255: POKES+13,255: REM
SUSTAIN/RELEASE TO MAXIMUM
90 POKES+3,8: POKES+10,8: REM SHAPE
OF PULSE
100 POKES+4,65: POKES+11,65: REM
TURN ON PULSE WAVEFORM
110 F=F+1: REM INCREMENT FREQUENCY
VARIABLE
120 IFF=36THENF=16: REM CHECK FOR
UPPER FREQUENCY LIMIT
```

Turn On a Waveform

Last, but certainly not least, the sound needs a waveform. The 64 and 128 both feature four waveforms, each with a characteristic timbre. The triangle's sound is soft and mellow, the sawtooth mimics a saxophone, the pulse is hollow, and the noise is, well, noisy.

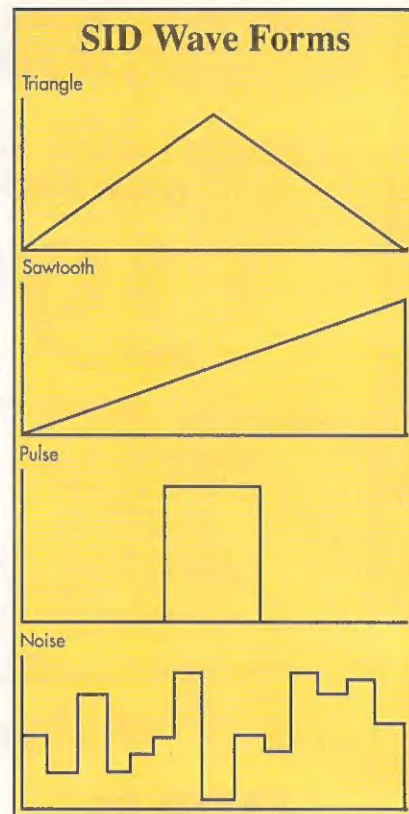
To actually begin the sound, we use voice 1's control register, S+4. We usually poke one of four particular values to produce the desired waveform.

Triangle	17
Sawtooth	33
Pulse	65
Noise	129

Here's the way we'll select a waveform in our program. For this example, let's select a triangle waveform and poke its value into S+4.

60 POKES+4,33: REM TURN ON SAWTOOTH WAVEFORM

I like waveform 33, the triangle; it has a nice bite to it. If you've been entering the lines as presented, you can now run the program. You should be rewarded with a nice strong note that begins suddenly and gradually dies out. (Be sure to turn up the vol-



ume on your TV or monitor. The 15 that we poked to 54296 ensures that a good strong signal leaves the computer, but it won't be heard if your monitor volume is too low.)

Six lines to create a sound; that's not too bad, is it? Just remember the order.

1. Clear the chip (S through S+24).
2. Turn up the volume (S+24).
3. Wait.
4. Set voice 1's frequency (S+1).
5. Set voice 1's envelope (S+5).
6. Turn on voice 1's waveform (S+4).

Other Registers

We produced sound with only three of voice 1's memory registers; we didn't use registers S, S+2, S+3, and S+6. Let's look at them now.

S is the register that fine-tunes voice 1's frequency, which was coarsely set with S+1. If you wanted just a noise or a beep of no particular frequency, S+1 would be enough frequency control. To accurately produce musical notes, however, we must also poke a value to S.

What value? For frequencies of musical notes, the values are listed in your User's Guide in a table appropriately called Music Note Values. For nonmusical sounds, such as drums, it's mostly a matter of trial and error. Let's fine-tune the frequency we poked into S+1 in line 40. Add this line to the program to give us an exact

130 POKES+1,F: POKES+8,F*1.3: REM CHANGE FREQUENCIES BOTH VOICES 140 GOTO110

We're using the variable F (defined in line 20) instead of the number 16 for the coarse frequency. The coarse frequency pops up first at line 50. In line 130, voice 2's frequency is calculated as a multiple (1.3 times) of voice 1's. Why? We do it to keep the interval between the two voices' frequencies roughly constant for a more authentic siren sound.

In line 110, we increase variable F by 1. Try different increments or try decreasing, instead of increasing, F. Line 120 limits the upper value of F. Try other limits or values less than 16 if you're decreasing F.

Once a limit is reached, F is reset to its original value. Line 130 once again pokes new values to both voices' frequency registers. Voice 1 gets newly increased F; voice 2 gets a multiple of F. Line 140 sends control back to line 110, which increases F again. The program stays in a loop from line 110 through line 140 until you stop it by pressing the Run/Stop key.

Only \$24.95



FOR THE C64 AND C128 IN 64 MODE

Fun Graphics Machine

FUN GRAPHICS MACHINE (FGM) IS AN "ALL-IN-ONE" GRAPHICS PROGRAM FOR THE C-64. WHAT CAN BE CREATED WITH FGM IS ONLY LIMITED BY YOUR IMAGINATION. JUST A FEW EXAMPLES:

SUPPORTS ALL CMD DRIVES	BUSINESS CARDS CUSTOM LABELS VIDEO TITLING NEWSLETTERS CALENDARS DIAGRAMS POSTERS FORMS	SIGNS CHECKS OVERLAYS BROCHURES LETTERHEADS CERTIFICATES GREETING CARDS DISK ENVELOPES	WORKS WITH 1541, 1571 & 1581 DRIVES
SUPPORTS MULTIPLE DRIVES	KORLA GEOPRINT PRINT SHOP ADV OCP ART VIDEO BYTE II	HANDYSCANNER 64 COMPUTER EYES PRINTERMASTER NEWSROOM GEOWRITE DOODLE	THIS AD CREATED WITH FGM

GEOS SCREENS CAN BE CAPTURED SIMPLY BY RESETTING COMPUTER THEN LOADING THE FUN GRAPHICS MACHINE.

FUN GRAPHICS MACHINE FULL KEYBOARD OVERLAY ---\$3.50 EA
PLEASE STATE COMPUTER (C64, C128, SX-64) OR C64 IS SHIPPED

FUN GRAPHICS MACHINE DEMO DISK THIS IS A PREVIEW OF WHAT FUN GRAPHICS MACHINE IS ALL ABOUT ---\$2.00

FOLLOWING DISKS REQUIRE THE FULL BLOWN VERSION OF FGM
FGM FONTS OVER 90 FONTS IN FGM FORMAT ---\$5.00
FGM CLIP ART VOL.1 OVER 200 EXCELLENT GRAPHICS ---\$8.00
FGM CALENDAR TEMPLATE DAILY, WEEKLY, MONTHLY ---\$5.00
FGM OVERLAY TEMPLATE MAKE FULL KEYBOARD OVERLAYS
STATE COMPUTER FOR OVERLAY TEMPLATES OR C64 IS SENT ---\$5.00
FGM UPDATE DISK V6.4 UPDATES FGM V6.X TO FGM V6.4 ---\$2.00

PLEASE ADD FOR SHIPPING AND HANDLING PER ORDER ---\$3.50
FOREIGN ORDERS: FOR AIR MAIL ADD ADDITIONAL AS FOLLOWS:
CANADA/MEXICO \$1.00, AUSTRALIA \$10.00, ALL OTHERS \$5.00
U.S. FUNDS ONLY SORRY NO CHARGE CARDS

The FGM Connection, P.O. Box 2206, Roseburg, OR. 97470
FOR MORE INFORMATION CALL 503-673-2234

New * Big Blue Reader 128/64 - 4.0

Transfers word processing, text, ASCII, and binary files between C64/128 and IBM PC compatible 360K 5.25" and 720K 3.5" disks.

New Version 4.0 features: Transfers ASCII, PET ASCII and Screen Code files including: WordWriter, PocketWriter, SpeedScript, PaperClip, WriteStuff, GEOS, EasyScript, Fleet System and most others. Supports drives # 8-30. New Backup (C128) and Format (1571/1581) programs. Reads MS-DOS sub-directories, uses joystick, and more.

Includes C128 & C64 programs. Requires 1571-or-1581 Disk Drive.

Big Blue Reader 128/64 - 4.0 only \$44.95

Version 4.0 upgrade, send original BBR disk plus \$18.

Bible Search 3.2

1. Complete Old & New Testament text on (4) 1541/71 or (2) 1581 disks.
2. An Exhaustive English Concordance on (2) 1541/71 or (1) 1581 disks. Includes more than 700,000+ references.
3. Incredible five (5) second look-up time, per/word, per/disk.
4. Instant, automatic spell checking of more than 12,800 words.
5. Wildcard and Boolean AND, OR & NOT search options.
6. Search the entire Bible in 5 seconds with 1581 or HD (version 3.52).
7. Money Back Guarantee!

Features: C64 & C128 programs; screen, printer and disk output; users guide, disk case. Available on (7) 1541/71, or (4) 1581 disks.

KJV \$49.95 | NIV \$59.95 | Both \$90

Any questions? Call or write for more information.

Order by check, money order, or COD. US funds only.

FREE shipping in US. No Credit Card orders.

Canada & Mexico add \$4 S/H, Overseas add \$10 S/H (\$5 BBR)

SOGWAP Software (219)724-3900

115 Belmont Road; Decatur, Indiana 46733

SAVE TIME AND MONEY

Yes, save time and money! Subscribe to the *Gazette Disk* and get all the exciting, fun-filled *Gazette* programs for your Commodore 64 or 128—already on disk!

Subscribe today, and month after month you'll get all the latest, most challenging, and fascinating programs published in the corresponding issue of *COMPUTE*.

New on the *Gazette Disk*! In addition to the programs that appear in the magazine, you'll also get outstanding bonus programs. These programs, which are often too large to offer as type-ins, are available only on disk—they appear nowhere else.

As another *Gazette Disk* extra, check out

"Gazette Gallery," where each month we present the very best in original 64 and 128 artwork.

So don't waste another moment. Subscribe today to *COMPUTE's Gazette Disk* and get 12 issues for only \$49.95. You save almost 60% off the single-issue price. Clip or photocopy and mail completed coupon today.

Individual issues of the disk are available for \$9.95 (plus \$2.00 shipping and handling) by writing to *COMPUTE*, 324 West Wendover Avenue, Suite 200, Greensboro, North Carolina 27408.

YES! Start my one-year subscription to *COMPUTE's Gazette Disk* right away for only \$49.95.*

☐ Payment enclosed (check or money order)

☐ Charge ☐ MasterCard ☐ Visa

Acct. No. _____ Exp. Date _____

Signature _____ (Required)

Name _____

Address _____

City _____

State/ZIP/ _____

Province _____ Postal Code _____

Mail to *COMPUTE's Gazette Disk*, P.O. Box 3250, Harlan, IA 51593-2430

* Residents of NC and NY, please add appropriate sales tax for your area. Canadian orders, add 7% goods and services tax.

pitch of middle C on the piano.

45 POKES,195: REM FINE-TUNE FREQUENCY

Shaping the Pulse

While S+2 and S+3 control the shape of voice 1's pulse waveform, S+2 is rarely used. Poking a value of 8 to S+3 will give the pulse waveform a nice, even shape. It's not necessary, however, to shape a pulse waveform unless you plan to use it. To hear what the pulse sounds like, add line 55 and change line 60 as follows.

55 POKES+3,8: REM SHAPE OF PULSE

60 POKES+4,65: REM TURN ON PULSE WAVE FORM

Run the program again, and listen to the difference in the sound. Now experiment. Try waveforms 17 (triangle) and 129 (noise). Try various frequencies and envelopes. A reminder: Don't confuse voices with waveforms. SID has three voices (remember our three-string guitar?) and four waveforms (triangle, sawtooth, pulse, and noise).

Voices 2 and 3

So much for voice 1. If you want to play more than one voice at a time, each must be set up independently.

For instance, let's add another note to harmonize with the last one. Modify lines 40-60.

40 POKES+1,16: POKES+8,21

45 POKES,195: POKES+7,31

50 POKES+5,12: POKES+12,12

55 POKES+3,8: POKES+10,8

60 POKES+4,65: POKES+11,65

Voice 2's values follow the colon in each line. To program voice 2, just add 7 to voice 1's memory registers. In line 40, S+1 for voice 1 becomes S+8 for voice 2; in line 45, voice 1's S becomes voice 2's S+7; and so on.

Notice that in this example I've poked all voice 2 registers with the same values—except frequency in lines 40 and 45. Frequency values 21 and 31 (from the Music Note Values table) are needed to produce E above middle C on the piano. You may, if you like, set different envelopes for each voice (line 50) or different waveforms (line 60). If you run the program now, you'll hear a two-note chord in perfect harmony.

As you've probably noticed by now, SID's three voices are arranged within the chip in groups of seven registers each. Thus the control registers for voices 1, 2, and 3 are 54276, 54283,

and 54290, respectively. The attack/decay portion of the three envelopes is set in registers 54277, 54284, and 54291, respectively. Therefore, to program voice 3, just offset the memory registers by 7 again.

As promised, here's how to reduce the proliferation of POKES for this particular program. This technique won't always be applicable, but it may give you some ideas. Begin by copying lines 10 and 30 from the above program. Then delete the remaining lines. Now add these lines.

40 FORG=1 TO 10: READL,D:POKES+L,D

50 NEXT:END

**100 DATA 1,16,8,21,0,195,7,31,
5,12,12,12,3,8,10,8,4,65,11,65**

That's it! All SID's offsets from S (54272) and the pokable values have been compressed into one data line. One FOR-NEXT loop does the rest of the work.

While this simple program touches on only a few of the SID chip's wonderful possibilities, you can have fun experimenting with changing waveforms, frequency values, and voices. I hope programmers will be encouraged to further explore the sound capabilities of their 64s and 128s. □



**8 BIT**

PO BOX 542

LINDENHURST NY 11757-0542

DON'T LET A SPILL CAUSE A COMMODORE**6 PAC SETS**

EACH SET \$ 5.00

1: ASST. (Star Trek+)

2: EDUCATIONAL

4: GAMES (Tetris+)

5: DEMOS/MOVIES

6: PRODUCTIVITY

7: GEOS CLIP ART

8: RECIPES SET

9: UTILITIES SET

A: CHRISTMAS

B: ASST. (Mario+)

C: MUSIC SET

D: ART GALLERY

E: GEOS FILES

SIX PAC #2 \$ 10.00

ADULT IMAGES

You must sign that you

are over 18 years old

to receive this set!

MINDSCAPE
HANDGRIP
JOYSTICK
ONLY \$5.00

MELTDOWN**ONE SMALL MISTAKE COULD DESTROY YOUR COMMODORE COMPUTER FOREVER**

The finest quality keyboard seals are available for your Commodore computer. Custom molded for each model, these seals fit over every key so precisely that you won't even know it's there... Until you spill something, that is!

Keep out accidental spills, dirt, dust, and ash! Never Again Wear Off The Print On Your Keys!

Save Your Commodore for **ONLY \$19.00**

Order # VS64fits your Commodore 64 or Vic 20

Order # VS64C.....fits your Commodore 64C

Order # VS128.....fits your Commodore 128

COMMERCIAL SOFTWARE!

CALL FOR ITEM AVAILABILITY!

DIE HARD\$ 5.00

BLOCKOUT\$ 5.00

PARADROID\$ 5.00

CLUBHOUSE SPORTS\$ 5.00

SEGA ARCADE 5 PACK.....\$ 15.00

(Includes Out Run, AfterBurner, Shinobi, Alien Syndrome, and Thunderblade!)

MANY ITEMS ARE CLOSEOUT ITEMS ONLY, AND THE AMOUNT OF STOCK MAY BE LIMITED! CALL OUR OFFICE TO VERIFY AVAILABILITY!

SHIPPING \$2.00 For First Item + \$1.00 each additional item

***U.S. Funds Only! *SORRY NO C.O.D.'s OR CREDIT CARD ORDERS**

FOR A FREE COPY OF OUR CATALOG, CALL:

(516)-957-1110 MONDAY - FRIDAY 10 am to 5 pm EST

Circle Reader Service Number 162

Create a
 Western
 Style
 with your
 Commodore
 64/128
 Computer

WESTERN HERITAGE

\$24.95

plus \$1/H
 US\$ 4.00
 Canada.....\$ 9.00
 AK, HI\$ 6.00
 AK, HI, PO.....\$ 6.00
 WA, Pac. Tix \$ 1.92

For more info,
 509-278-6928
 No C.O.D.'s

WESTERN HERITAGE

Graphics, Borders, and Fonts for the Print Shop.



HORSE FEATHERS
GRAPHICS
 Version for the Commodore 64/128 includes 9 Pins and 7 Pin Printers.

Requires:

Print Shop
 Version 2.0
 or

Print Shop
 Version 1.0
 with the

Companion
 or

Print Shop
 Version 1.0
 with the

Holiday
 Edition



Print Shop,
 Companion, and
 Holiday Edition
 are trademarks of
 Broderbund
 Software Inc.

Over 140 New Ways to Create a Total Western Environment with 90 Graphics, 42 Borders and 10 Fonts for the Print Shop.

- Create Western Style Stationery, Cards and Invitations.
- Invite Your Friends to a Western Birthday Party, Bar-B-Q or Card Game.
- Make 10 Gunfighters of the Old West, Wanted Posters.
- Impress Your Club with Western Posters, Banners and Calendars.
- Make posters for Your Favorite Western Event, Horse Show, Hay Ride.
- Designs for over 50 Western Business Activities and Club Events.
- Kids Share Secret Messages with Your Friends.



Horse Feathers Graphics Inc., N. 27310 Short Road, Deer Park, Wa. 99006-9712

Circle Reader Service Number 234

The GRAPEVINE GROUP Inc.

COMMODORE UPGRADES

SPECIALS

- **COMPUTER SAVER** This C-64 Protection System saves you costly repairs. Over 52% of C-64 failures are caused by malfunctioning power supplies that destroy your computer. Installs in seconds between power supply & C-64. No soldering. 2 year warranty. An absolute must and great seller **\$17.95**
- **PRINTER PORT ADAPTER** by Omnitronix. Avoid obsolescence. Allows you to use any Commodore (C-64) printer on any PC compatible or clone. Does not work with Amiga. **\$34.95**

512K RAM EXPANDERS

By special arrangement with Commodore, we are able to purchase at a fantastic price 400 of the original 512K 1750 RAM expander units for your C64 or C128 computer. Now keep up with the latest technology. Upgrade to 512K with a simple plug-in module. Completely compatible and comes with software. If you have a C64 you will need a heavier power supply (4.3 amp), which we will give you for \$31.00. C128 users do not need this power supply. This is the original Commodore unit with over 800,000 sold..... **\$99.95**
 Super 1750 REU Clone (512K). Does not require a larger power supply **\$142.50**

COMMODORE DIAGNOSTICIAN II

Originally developed as a software package, then converted to a readable format, the Diagnostician has become a fantastic seller. With over 38,000 sold worldwide, Diagnostician II utilizes sophisticated cross-reference grids to locate faulty components (ICs) on all C-64 and C1541 computers (C-128/64 mode). Save money and downtime by promptly locating what chip(s) have failed. (No equipment of any kind needed.) Success rate from diagnosis-to-repair is 98%. Includes basic schematic **\$6.95**
 (Available for Amiga computers with 3 1/2" disk at **\$14.95**)

NEW POWER SUPPLIES

- A super-heavy, repairable, "not sealed" C-64 power supply with an output of 4.3 amps (that's over 3x as powerful as the original). Featuring 1 year warranty, ext. fuse, schematics, UL approved..... **\$37.95**
 (Includes bonus Commodore Diagnostician II (valued @ \$6.95))
- **Our Biggest Seller** • 1.8 amp repairable heavy duty supply for C-64. (Over 120,000 sold)..... **\$24.95**

REPLACEMENT/UPGRADE CHIPS & PARTS

6510 CPU.....	
6526 CIA.....	
6581 SID.....	
6567 Video.....	
PLA 906114.....	
A1: 901/225-6-7-9.....	\$9.95 EACH
4164 (C-64/RAM).....	60
C-128 ROMs Upgrade (set 3).....	24.95
C1571 ROM Upgrade (310654-05).....	\$10.95
C-64 Keyboard (new).....	19.95
C-64 Cabinet (new).....	\$49.95
Interface Cables #690 C64 to 1541/1571 disk drive.....	\$12.95
#693 C64 to 3 pin RCA (eg 1084).....	\$16.95
1541/1571 Drive Alignment.....	\$21.95
Super Graphics by Xetec.....	\$59.50
Service Manuals for C64, C128, 1802, 1084SP, 1541.....	\$21.95

EMERGENCY STARTUP KITS

Save a lot of time and money by repairing your own Commodore computer. All chips are direct socket plug-ins (no soldering). Each kit includes all you need to "start up"/revive your broken computer. Originally blister packed for the government PXs worldwide, this series is now available to you. Total cost savings per kit far exceeds purchasing chips on an individual basis.

KIT #3 (Part #DIA 15) for C64

Symptoms: No power up • Screen lock up • Flashing colors • Game cartridge problems
 Contains: ICs #PLA/82S100/906114, 6526, Commodore Diagnostician, Fuse, Chip Puller, 8 RAMs, Schematic, Utility Cartridge & special diagnostic test diskette with 9 programs
An \$87.50 value for only \$29.95

KIT #4 (Part #DIA 16) for C64

Symptoms: Control Port • Sound • Keyboard • Serial device problems
 Contains: ICs #6526, 6581, 8 RAMs, Commodore Diagnostician, Fuse, Chip Puller, Basic Schematic, Utility Cartridge & special diagnostic test diskette with 9 programs
A \$79.80 value for only \$29.95

KIT #5 (Part #DIA 17) for 1541/1571

Symptoms: Drive runs continuously • Motor won't stop • Read errors • No power up
 Contains: ICs #6502, 6522, Fuse Chip Puller, Basic Schematic, Commodore Diagnostician & special diagnostic test diskette with 9 programs
An \$70.10 value for only \$29.95

Send For Free Catalog
 3 CHESTNUT ST., SUFFERN, NY 10901
 Order Line 1-800-292-7445 Fax 914-357-6243
 Customer Service: 914-368-4242 International Order Line: 914-357-2424
 We Ship Worldwide Prices subject to change
 Hours: 9-6 E.T. M-F 15% Restocking Charge

Tell a friend you've heard it through the Grapevine.

Circle Reader Service Number 145

ARACHNOPHOBIA

A deadly spider from South America has migrated north, laying her loathsome eggs in hundreds of homes, schools, buildings, barns, and cemeteries. In a frighteningly short time, her offspring have hatched and have begun to reproduce.

Thus begins the arachnids' reign of terror in communities across America. This Disney arcade game for the 64 closely follows the basic premise of the studio's hit movie *Arachnophobia*.

Homes are overrun, citizens terrorized, and whole communities abandoned. Residents have tried everything to rid themselves of the unwanted guests, but nothing seems to stop these creepy crawlers. The eight-legged enemy is upon us. It's enough to make your skin crawl.

As a last resort, the U.S. Department of Agriculture sends a frantic plea to Delbert McClintock, owner of the McClintock Infestation Management Company. McClintock is the inventor of a patented insecticide, Toxi-Max, which is said to be strong enough to kill the arachnids. Fearless Delbert loads his bugmobile with the lethal Toxi-Max and a supply of bug bombs, and sets out to free his country from the invading horde.

You won't need a lot of practice to get into the swing of playing this game, nor will you need to refer to the instruction manual throughout play. Disney does recommend that you make a backup of the game's double-sided disk before playing and use the backup for play. The game is compatible with most Fast Load cartridges, too. Since there's enough variety

in *Arachnophobia*'s sharp, colorful graphics to keep you playing for hours, you'll find using a Fast Load cartridge will save you a lot of time, since you must flip sides during the game.

When you load *Arachnophobia*, you'll see the bugmobile as it drives past homes,

sense of timing and your joystick skills.

Spider webs are a real nuisance. Blundering into one will slow you down to half speed and make you more vulnerable until you break free.

A single spray of deadly Toxi-Max is enough to kill a

tant, the homemade flamethrowers can clear an entire floor or ceiling of a room with just one pass.

When you've cleared a structure, you can safely return to the bugmobile. But there's no time for you to rest. The battle has only begun! There are more buildings and towns needing your bug-slaying skills. Just guide your bugmobile to another building and start exterminating spiders.

Every building in every town is filled with hordes of vicious spiders, defending an egg sack. Only one structure in each town hides a queen spider. The queen is the same size as the original South American spider. You'll know this mean mama by the distinctive yellow markings on her legs. Watch out! She's tougher than her soldiers and can even bite after she's been stunned. Slaying her will transport you to another city with yet another queen spider to roust.

All in all, this is a challenging game that's designed to give you a real workout. If you succeed in besting the queen spider in every town, you'll have saved the country and proved yourself a hero. As a reward, the United Nations will give you a secret assignment in the Amazon rain forest. The monstrous arachnids there will make you wish you'd been a little less successful.

To aid you during play, the bottom of the screen displays status information. There's an amusing picture of Delbert that monitors the state of your health. It changes from smiling to frowning to screaming in pain, depending on how many times you've been bit. First-aid kits will restore Delbert's smile.

Next to Delbert's picture



Delbert McClintock is the nation's last line of defense against hordes of invading spiders from South America

farms, schools, and cemeteries. Pick the building you want to enter; then use your joystick to guide the bugmobile there.

When you stop at a location, the screen changes to an interior scene showing Delbert. Your mission is to help him clear the infested rooms by hunting down and destroying all the spiders and the egg sack that's hidden in every structure. Sound easy? Don't be so sure.

Spiders are everywhere. They'll do all they can to guard their egg sack. Sneaky ones drop from the ceiling to land on you; others slither down web strands and bite you from behind. You'll be attacked at ground level, too. Often, the soldier spiders work in groups, testing both your

soldier spider within spraying distance. It only takes a few spider bites to slay you, however, so keep your eyes open for first-aid kits. These will restore your strength. There's at least one kit in every building.

Don't forget you're wearing heavy work boots, too. You can stomp on some of the creepy crawlers, conserving your limited supply of insecticide.

The quickest way to clear a room is to use a bug bomb. You start the game with only three of these, so use them wisely. Other items you find as the game progresses are almost as useful as bug bombs. Matches and aerosol cans can be fashioned into nifty flamethrowers, which have a better range than your insecticide sprayer. More impor-

The Gazette Productivity Manager

(Formerly PowerPak)

Harness the productivity power of your 64 or 128!

Turn your Commodore into a powerful workhorse, keep track of finances, generate reports in a snap, manage your money in minutes—all with the new 1991 *Gazette Productivity Manager*! Look at all your 64/128 *Productivity Manager* disk contains.

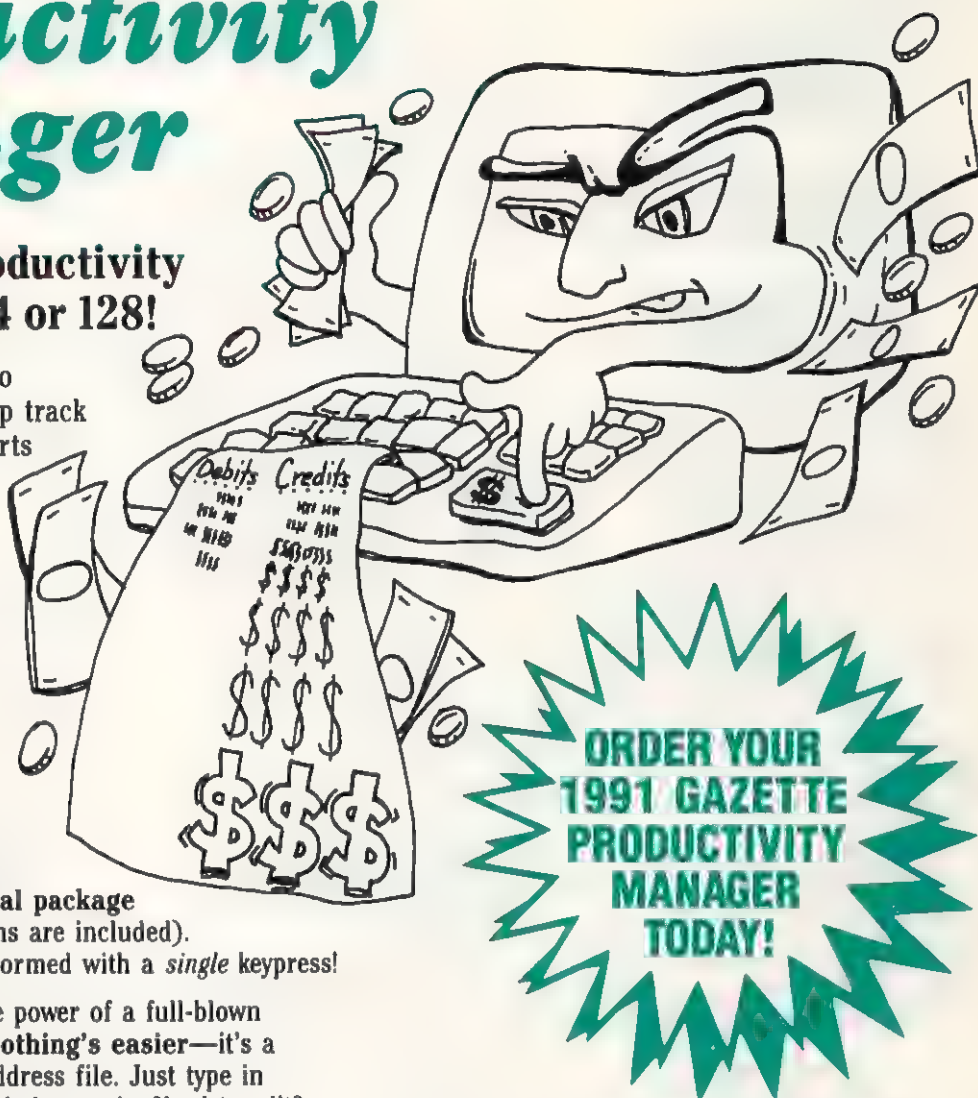
GemCalc 64 & 128—

A complete, powerful, user-friendly spreadsheet with all the features you'd expect in an expensive commercial package (separate 64 and 128 versions are included). Most commands can be performed with a *single* keypress!

Memo Card—Unleashes the power of a full-blown database without the fuss! **Nothing's easier**—it's a truly simple computerized address file. Just type in your data on any one of the index cards. Need to edit? Just use the standard Commodore editing keys. Finished? Just save the data to floppy. What could be easier?

Financial Planner—Answers all of those questions concerning interest, investments, and money management that financial analysts charge big bucks for! You can plan for your children's education and know exactly how much it will cost and how much you need to save every month to reach your goal. Or, decide whether to buy or lease a new car. Use the compound interest and savings function to arrive at accurate estimates of how your money will work for you. **Compute the answer at the click of a key!**

DON'T MISS OUT ON THIS POWERFUL WORKHORSE!



(MasterCard and Visa accepted on orders with subtotal over \$20).

☐ **YES!** Please send me ____ *Productivity Manager* disk(s) (\$14.95 each).

____ Subtotal

____ Sales Tax (Residents of NC and NY please add appropriate sales tax for your area. Canadian orders, add 7% goods and services tax.)

____ Shipping and Handling (\$2.00 U.S. and Canada, \$3.00 surface mail, \$5.00 airmail per disk.)

____ Total Enclosed

____ Check or Money Order ☐ MasterCard ☐ VISA

Credit Card No. _____

Signature _____ (Required)

Daytime Telephone No. _____

Name _____

Address _____

City _____

State/Province _____ ZIP/Postal Code _____

Send your order to Gazette 1991 Productivity Manager, 324 W. Wendover Ave., Ste. 200, Greensboro, NC 27408.



1991 Gazette Index

Everything's included!
Features, games, reviews,
education/home applications,
programming, bugswatter,
feedback, and columns!

A superb interface includes pull-down menus, help screens, and keyboard, joystick, or mouse control. Features include super-fast searching and sorting capabilities. An options screen allows you to choose text colors, drive number, and input device. And there's full documentation on disk.

Choose from three modes of operation—*browse* for quick scanning, *view* for detailed information and descriptions, and *edit* for adding items from upcoming issues—and print to any printer. There's even a turbo-load option for maximum disk-access speed.



To order, send \$7.95 per disk, the quantity of disks ordered, check or money order,* your name and complete street address:

**1991 Gazette Index
324 West Wendover Avenue
Suite 200
Greensboro, NC 27408**

* Please add \$2 shipping & handling (\$5 foreign) for each disk (residents of NC, NJ, NY please add applicable sales tax; Canadian orders, add 7% goods and services tax).

All payments must be in U.S. funds. Please allow 4 weeks for delivery.

REVIEWS

is an indicator showing how much Toxi-Max insecticide remains in your spray tank. It refills automatically whenever you return to the bugmobile. (Beware! You can leave a building at any time for refills, but all the spiders will return while you're out.)

A counter at the bottom of the screen shows how many bug bombs you have. You start with three but earn another every time you clear a structure. Last, but definitely not least, is the Bugometer. This compasslike device indicates the direction in which to travel to find the queen in each city.

I prefer to play Arachnophobia with the sound turned down, but my son likes to hear Delbert's bug-fighting comments and the sizzle of his flamethrower. I like to go through houses systematically rousting arachnids; he zeroes in on the queen. Even though our styles vary, however, we both agree that this is the best arcade game we've played in a long while.

MARTI PAULIN

Commodore 64 or 128—\$29.95

WALT DISNEY COMPUTER SOFTWARE
500 S. Buena Vista St.
Burbank, CA 91521
(818) 841-3326

Circle Reader Service Number 341

FUN GRAPHICS MACHINE

Fun Graphics Machine is a great way to create and manipulate graphics and hi-res screens on a 64. My introduction to the program was a free demo that's available on QuantumLink. I was amazed at the ways that I could work with the designs on the screen: flip, flop, reverse, stretch, shrink, crop, rotate, fasten, copy, and print the results. The demo won't allow you to save and print features, but the asking price for the registered version makes it a real must-have bargain. The reference manual is the first good feature.

The spiral-bound manual lies flat, so you can really use it. Some of the instructions are duplicated, but that stops the page flipping when you need to refer back to some detail that has slipped your memory.

The manual itself was produced with FGM. It even shows step-by-step instructions of how various pages were composed. This is not a drawing program. It doesn't have lines, circles, or squares, and there are no colors—just a white screen (or rather, three white screens) to work with.

The program uses color in a useful way. The cursor and borders change color to let you know what mode you're in. Blue cursor is text, gray is grab, purple is adjust, and so on.

You work on a 40-column screen, and the screens can be linked both across and down. By combining two screens across, you have your 80-column format for printing a full page. You can use a third screen as a workplace. Link the screens down for as many as you need. Print a banner of any length.

Save your work with a simple method of coding, and then use one instruction to print all of your work.

FGM is really a collection of programs, not just one. With the program disk in one drive, it'll recognize the presence of any other two drives. Create lets you do your own thing. Demo runs demos that are available on the program disk or replays those that you create and save. Clip-Art uses artwork found in other programs like The Print Shop, The Newsroom, and Doodle. Printer sends your work to your printer.

You can print your work to disk in files that others can view without having to run FGM. You can design and send greeting cards or draw screens to be used as titles on your VCR.

FGM has its own department on Q-Link. Download a file with 50 different fonts, and you can type in just about any style that you like. FGM contains a customizer, which will design or modify a font. Updates are always being added. If you have a question, someone online will have the answer, and samples of what users have done are always interesting to study.

If you're using a 128 and have the reset switch, you'll discover something remarkable. Suppose you're running a program in 64 mode and see a graphic on the screen that you'd like to save. Hit the reset switch. The program will be gone, but then load and run FGM. On most occasions the graphic will be available on one of FGM's screens. Now you can save it, grab part of it, and paste it on another screen. Have fun; that's what the program is all about—having fun with graphics.

Since you can edit at the pixel level, you can do some finely detailed work, and a smoothing technique takes away some of the rough spots on captured pictures. With the overlay method of grabbing and pasting, you can design and save different templates and then use them for various projects. A simple template with a musical symbol font and score lines is useful for writing musical scores. A grid pattern can be overlaid with needlework designs.

Playing with FGM can become addictive. Searching for different graphics to manipulate, adding new eyes to a face from a Print Shop cartoon, using part of a picture as the cover for a greeting card, and designing your own letterhead are just some of the fun you can have working with graphics.

In text mode you can link two screens across, use word-wrap, select a font, set the margins, and type your document. The size of the cursor can be changed with a single keypress. And with that size change, the size of your font changes, all the way to a full screen.

Great graphic work on the screen can be work wasted if you can't send it to a printer. FGM supports most printers, and it gives you the option of telling your printer to perform various effects. You can select dots-per-inch density; single or double height; single, double, or triple width; various margins; and so on.

Try printing the same screen with different options, and you'll be surprised by the results. Not only does FGM let you design and work with your own graphics, but you have the ability to load files from other programs. The possibilities are endless. You have complete control over every pixel on the screen. Artwork can be stretched, shrunk, slanted, rotated, overlaid with shadows, and more. By using two screens and flipping between them, you can create simple automation for your demos.

Learning to use the program can take time, but you don't have to learn it all at once. If you go too far, a couple of keystrokes will always take you back to where you started. There's no need to remember filenames.

Selections are made from a screen menu, and a disk directory is always available. You can use up to three drives with FGM, and the program will ask you which one you want to access. You can customize your program disk so that it will default to your particular printer.

If you'd like the cursor and borders to be different colors, you can change them. Copy the program disk and then customize the copy with your most used fonts, character sets, and graphics for a program default to suit your own needs.

FGM is always being updated on Q-Link. The author, whose Q-Link handle is RonH8, is often online in the Starving Artists' Cafe. He is always offering new hints and suggestions.

Q-Link members can download an FGM demo and try it before buying. But once you try FGM, you'll be hooked on graphics—and spoiled. No

more having a graphic that won't fit in the space you need on your document. With FGM you can copy it, shrink it, expand it, paste it, and then smile at the results.

Discover that your 64 is a real fun machine. Then surprise your friends with your newly discovered artistic talent. You won't go wrong with Fun Graphics Machine.

ESTHER OLSON

Commodore 64 or 128—\$24.95, plus \$3.50 shipping and handling

THE FGM CONNECTION
P.O. Box 2206
Roseburg, OR 97470
(503) 673-2234

Circle Reader Service Number 342

DWEEZILABEL

If Dweezil is anything like the program that bears his name, he must be one clever dog. Once again, Dave Ferguson, GEOS programmer *extraordinaire* and human who lives with Dweezil, has released an intriguing and useful GEOS program with a picture of a dog in a party hat on the label.

DweezilLabel is everything a GEOS user could want in a label program. Well, to be fair, it's everything Ferguson would want in a label program. He'll be the first to admit that the program evolved more as an answer to his specific needs than as a general-purpose label maker. Even so, it includes enough features to function as a minidatabase, a minipublisher, and who knows what else.

In the course of running Quincy Software, Ferguson needs to keep track of customers from all over the world and to keep notes on what they've ordered, how much they've paid, and so on. DweezilLabel emerged as his ideal multipurpose low-end business application. You can find it on Dweezil Disk #3, which includes MYgeoDIARY and geoWORDS.

Since Ferguson runs his business exclusively with GEOS products, data from DweezilLabel is compatible with applications such as geoMerge and geoCalc. Text scraps and numeric data can be neatly clipped in formatted chunks to fit those GEOS applications.

An even better example of DweezilLabel's versatility is the way it handles data. The program works with files of up to 50 records, similar to a card file database. These files can be created from within DweezilLabel, geoWrite, or geoFile. Ferguson wanted room in those records for more than just names and addresses. He wanted to keep notes about what products people had ordered and the amount of



Can Your Computer Make YOU \$1,000,000?

WITH LOTTERY PC YOUR NEXT TICKET COULD BE WORTH MILLIONS!

LOTTERY uses the raw power and storage of your computer to determine and refine the number selection methods that will win the various lottery games you play. Don't be limited to the one or two methods that other programs use, they might not work in your state. There is no better system available!

Join the growing list of winners using our system.

SPECIFY:
 Lottery 64(C64/128)
 Lottery PC
 IBM PC/XT/AT and compatibles

Commodore 64/128 & Plus/4 are registered trademarks of Commodore Int
 IBM PC/XT/AT are registered trademarks of International Business Machines Inc

To order, send \$29.95 for each plus \$3.00 postage & handling per order to
 (Illinois residents add 6% sales tax)
 (Orders outside North America add \$3.00)




C.O.D. orders call:
 (708) 566-4847

Superior Micro Systems, Inc.
 26151 N. Oak Ave.
 Mundelein, IL 60060



Circle Reader Service Number 221

DISKS O'PLENTY INC

7958 PINES BLVD. SUITE 270A
 PEMBROKE PINES FL 33024
 (305) 963-7750

Call or write for free descriptive catalog of
 C64/128 Public Domain & Shareware
 Choose from over 900 Disks
 Adult list of over 50 Disks available
 to those 18 or over.

Choose 6 for \$5.00 LIMITED OFFER	021MU	SID MUSIC UTILITIES
	019GR	PRINTSHOP UTILITIES
	019ED	JR HIGH EDUCATION
	062ED	HIGH SCHOOL EDUC.
	033ED	TYPING / SPANISH
	031ED	COMPUTER SCIENCE
	010UT	PIRATES TOOLBOX
	119GA	FOREIGN ARCADE
	022GA	CASINO-BOARD GAMES
	021GE	GEOS FONTS
	002MS	LOTTERY PROGRAMS
	003MS	COLLECTORS CORNER

Circle Reader Service Number 253

THE AMERICAN HEART
 ASSOCIATION
 MEMORIAL PROGRAM.



American Heart Association

This space provided as a public service.

money they'd paid, so he added several extra data lines for that express purpose, data that the labeler part of the program doesn't print unless you want it to. So far, that's pretty tame stuff, but this is no wimpy Rolodex.

Tucked away in the Text menu is a series of search commands that let you sail through your data with ease. The six possible lines of data could be names and addresses. You could store shoe sizes and a recipe for Pan Galactic Gargle Blasters in there if you wanted, but Dweezilabel restricts you to the number of spaces you can use. In fact, aside from the size limit and the lack of the trivial feature of saving a graphic to a record, Dweezilabel can hold its own with geoFile for usefulness. As I said before, it even creates merge files for geoMerge.

But, hey, what about labels? Yes, Dweezilabel does labels, any kind of labels. It produces any kind of printed output that is 2 inches tall, for that matter, on pages up to a full 11 inches tall. Using a technique called layering in the work window, you can put together combinations of graphics and text to create just about any kind of label you can imagine.

By paging through the database, you can select label text that can be modified however you like. You can use any GEOS font you might have available (on either disk, up to the file selector's limit—no six or seven font maximum here) and any style, including reverse. The work window is conveniently sized to fit Ferguson's premier graphics desk accessories, NewTools and geoStamp (available on other Dweezil Disks). This means you can stamp yourself a border around a label or curve and angle graphics and text to your heart's delight.

All this power doesn't come as easily as it could, however. The documentation provided on disk is extensive but a bit thin in spots. The entire process of layering a graphics label is not particularly intuitive, which is not necessarily bad, but a step-by-step tutorial for this process would save the user some trial and error.

The words *scrap* and *label* appear often, sometimes meaning one thing and sometimes another. While these variations are defined in the short glossary,

they do get confusing. Maybe since this program has become second nature to Ferguson, he's lost the perspective of a neophyte. The documentation should've been written from the perspective of the user who hasn't a clue about how this program operates—but it wasn't. As a result, this program runs the risk of being tossed aside after a half hour of frustration by casual users who don't care to figure out things on their own.

That would be a shame. Dweezilabel is too powerful an application to be missed by anyone who has some honest-to-goodness work to do with GEOS. The results are worth the extra effort it takes to master the intricacies of the interface. Heck, when used in conjunction with NewTools and geoStamp, Dweezilabel might be, as the ads claim, the "hottest GEOS label program to come along in years!"

STEVE VANDER ARK

Commodore 64 or 128, GEOS—\$15 95

QUINCY SOFTWARES
9479 E. Whitmore Ave
Hughson, CA 95326-9745

Circle Reader Service Number 343

TYPING AIDS

MLX, our machine language entry program for the 64 and 128, and *The Automatic Proofreader* are utilities that help you type in Gazette programs without making mistakes. To make room for more programs, we no longer include these labor-saving utilities in every issue, but they can be found on each *Gazette Disk* and are printed in all issues of *Gazette* through June 1990.

If you don't have access to a back issue or to one of our disks, write to us, and we'll send you free printed copies of both of these handy programs for you to type in. We'll also include instructions on how to type in *Gazette* programs. Please enclose a self-addressed, stamped envelope. Send a self-addressed disk mailer with appropriate postage to receive these programs on disk.

Write to Typing Aids, COMPUTE's Gazette, 324 West Wendover Avenue, Suite 200, Greensboro, North Carolina 27408.

ONLY ON DISK

In addition to the type-in programs found in each issue of the magazine, *Gazette Disk* offers bonus programs. Here's a special program that you'll find only on this month's disk.

BASSEM 128

By Fernando Buela Sanchez
Querétaro, QRO
Mexico

Symbolic label-based assemblers are the most convenient way to write machine language programs. You enter instructions as source code, and they are later assembled into object code. Rather than using memory locations, you can use meaningful labels.

Many programmers have used—and raved about—Bassem for the 64, and now there's an improved version for the 128. Bassem 128 works in conjunction with BASIC 7.0, and because of the 128's larger memory, it can store larger source code programs. With the addition of new commands, you can also develop your programs with less effort.

Bassem 128 and complete instructions are available only on disk. You can have this program and all the others that appear in this issue by ordering the November *Gazette Disk*. The price is \$9.95 plus \$2.00 shipping and handling. Send your order to *Gazette Disk*, COMPUTE Publications, 324 West Wendover Avenue, Suite 200, Greensboro, North Carolina 27408.

HELP PREVENT HEART ATTACK WITH A STROKE.



Any type of aerobic exercise program can help reduce your risk of heart attack and stroke. The only hard part is diving in. To learn more, contact the American Heart Association, 7272 Greenville Avenue, Box 47, Dallas, TX 75231-4596.

You can help prevent heart disease and stroke. We can tell you how.



This space provided as a public service.
© 1992, American Heart Association

DISK MAGAZINES FOR 64 & 128

Great programs & articles from both sides of the Atlantic.

C64 ALIVE! is U.S. produced. LIGHT DISK and dubLIGHT are UK produced.

C64 ALIVE! Sample disk \$3 (£1.50): 5 issues ending 12/92 \$20 (£10);

6 issues starting 1/93 \$25 (£12.50)

dubLIGHT Single issue \$5 (£2.20):

12 issues starting 9/92 \$50 (£23.40)

LIGHT DISK (only 4 issues) 8 Disks \$30 (£15)

LIGHT DISK and dubLIGHT are for 64/128 — C64 ALIVE! is 64 only

FOR DELIVERY:

IN U.S.: Jack Vander White, C64 ALIVE!, P.O. Box 232115, Sacramento, CA 95823

IN UK: Datasphere Publications, 7 Fallowfield Close, Valley Drive, Norwich, NR1 4NW
Outside North America and UK write for prices.

Circle Reader Service Number 154



GRAFIX GALORE

Original Printshop Graphics



Over 80 super graphics to add sparkle to your Printshop projects! Everything from sports to spys and pirates to pizza.

Send \$11.95 (inc. s/h) add \$3 if outside N. America. Specify C-64 or IBM version.

— REQUIRES PRINTSHOP OR GRAPHICS COMPATIBLE PROG. —

C-64

CLIP ART CUPBOARD

P. O. BOX 317774 • CINCINNATI, OH 45231

IBM

SEC CHECK REGISTER 128

Manages personal or small business checking in a fast/efficient manner. Fast data entry, many bank transactions predefined. Unlimited recurring payees. Up to 750 active file transactions. History files allow an on going record. Up to 999 reference (account) numbers. Easy editing with many powerful commands. Reports printed by, Outstanding Transaction, Transaction, Reference Number, Reference Number & Date, Date, Date & Random Reference Number, or Payee. Print any type of personal or form feed check. Supports all 15XX and Hard Drives. Compatible with all currently available DOS cartridges and ROM chips. Custom video fonts. Spiral bound lay flat manual and much more. System requirements: C=128 with 80 column RGB or Mono. monitor. FREE with each order SEC Financial Calc. 128 TO ORDER: Send Check or Money Order for \$24.95 + \$3.00 S&H to: SPARKS ELECTRONICS, 5316 So. 9th, St. Joseph, MO 64504-1802

Circle Reader Service Number 252

COMMODORE 64 PUBLIC DOMAIN

Highest Quality Since 1987*

Games, Education, Business, Utilities, GEOS, Music, Graphics & More. As low as 90¢ per collection. 1 stamp for complete catalog or \$2.00 for catalog AND 30 sample programs (refundable). 24 hour shipping.

64 DISK CONNECTION

4291 Holland Rd., Suite 562 • Virginia Beach, VA 23452
(* Formerly RVH Publications)

Circle Reader Service Number 254

KeyDOS ROM Version 2 is here!

The KeyDOS ROM is a chip for the empty socket inside your C128 that adds more than 40 powerful features. KeyDOS is available instantly as soon as you switch on your 128!

KeyDOS is loaded with useful tools to simplify file access on multiple drive systems without typing file names—all major DOS functions included. Select multiple files for copying, viewing, printing, renaming or scratching. ASCII/CBM/Screen code converter. Full support for 1581 subdirectories. Built-in RAMDOS for REUs up to 2MB. New GEOS SuperBoot
Alarm clock. Disk editor. Powerful debugger

Only \$32.50. Satisfaction Guaranteed! Write for more information.

Enhance your system with the speed and convenience that KeyDOS provides!

Antigrav Toolkit, PO Box 1074, Cambridge, MA 02142

Shipping outside of US, Canada and Mexico add \$3

Circle Reader Service Number 244

Questions and
answers
about computer
memory,
onscreen messages,
and more

More Memory

What exactly is the purpose of expanding the 64's memory, using cartridges such as the 1750? On an IBM, certain amounts of memory are required to use certain software. Is there any software for the 64 that requires more memory than the 64 has?

JOHN VEILLEUX
ORRINGTON, ME

There's no software that we know of which requires more memory of the 64 than what is native to the machine. On the other hand, several software packages, such as GEOS, can make use of RAM expansion if it's available. Many programs—games in particular—use the disk drive for virtual storage when either the program or its data is too large to be loaded and maintained in memory at one time. If more of the game can be stored in memory, then the game runs faster and the user doesn't have to wait for the computer to access the data stored on disk.

Large spreadsheets and databases are two reasons why business applications benefit from larger memories. Programmers can use more memory, which allows for code that is more highly developed and interpreters or compilers that are more sophisticated. More memory is also a boon to graphics, especially animation, where several scenes must reside in memory at once for smooth screen updates. A computer can do great things with digitized sound, but a lot of storage space is needed to contain reasonable sound samples.

Where speed isn't a critical factor, disk drives are a practical means of extending the 64's 64K limit. But where speed and quick responses are needed, more memory is very handy indeed.

Flashing Message

I've been working on some games for the 64 and have run up against a problem. There are certain messages, such as *DANGER*, that I'd like to have flash on the screen. How do I do this?

CAL BODWIN
GREENSBORO, NC

You could flash a message in BASIC by alternately printing in normal and reverse mode again and again. The program would have to stop while the message blinked, however. When the program continued, the flashing would stop.

Here's a machine language solution. The following program will flash in black any message that is printed on the screen. Other colors will print normally.

```
10 FOR A=828 TO 914: READB:
   POKEA,B: C=C+B: NEXT: IFC
   <>8545THENPRINT"DATA
   ERROR": STOP
15 POKE 6,0:SYS828
   :POKE53281,1: POKE53280,1:
   PRINT"[CLR]3 DOWN]15
   RIGHT]BLK]DANGER]"
20 DATA 120,169,81,141,20,3,
   169,3,141,21
30 DATA 3,169,0,141,147,3,
   141,148,3,88
40 DATA 96,206,148,3,16,58,
   169,10,141,148
50 DATA 3,169,0,133,2,133,4,
   169,4,133
60 DATA 3,169,216,133,5,162,4,
   160,0,177
70 DATA 4,41,15,197,6,208,9,
   77,2,41
80 DATA 127,13,147,3,145,2,200,
   208,236,230
90 DATA 3,230,5,202,208,227,
   173,147,3,73
100 DATA 128,141,147,3,76,
   49,234
```

If you want a different color to flash, poke its color code (0-15) into location 6. The speed of the flashing can be adjusted by poking location

855 with a number from 0 to 255; the smaller the number, the slower the flash rate. SYS 828 enables the flashing messages. To stop the flashing, press the Run/Stop key and tap the Restore key.

Sequential Files

Could you please explain what a sequential disk file is and how to create one?

JACK DEMEANOR
CHARLESTOWNE, MA

A sequential file provides a way of keeping information separate from the program that uses it. This allows you to create general-purpose programs that can act on different sets of information. Instead of writing one program to keep track of a stamp collection, for example, and a second program to list a collection of rare books, you could write (or buy) a general inventory program that stores data in sequential files. One file would contain notes about stamps, and another would have the data about the books.

A single program could handle two or more different files. Sequential files are like DATA statements because you start reading at the beginning and continue until the end.

To create a sequential disk file, open it for writing, write one or more pieces of information to it, and then close the file. It's important to close a file when you've finished using it; otherwise, some of the information will be lost.

Reading the file requires an operation similar to that for writing. Open the file for reading, read the information, and then close the file.

Here's a short program that creates a sequential file.

```
10 PRINT "ENTER THREE
   NAMES"
20 PRINT"(PRESS RETURN AFTER
```


EACH ONE"
**30 PRINT"OR SEPARATE THE
 FIRST TWO WITH COMMAS)"**
40 INPUT A\$,B\$,C\$
50 OPEN 1,8,2,"NAMES,S,W"
60 PRINT#1,A\$: PRINT#1,B\$:
PRINT#1,C\$
70 CLOSE1

The three numbers after the OPEN command in line 50 are the logical file number, the device number, and the channel. The file number can be any number that's not already being used by a peripheral. If you had previously opened a file to printer with OPEN 1,4 (file 1, device 4), you couldn't use logical file number 1 for opening the disk file. The logical file number is important because it's the number used to read from and write to a file.

The second number after OPEN is the device number (a single disk drive is device 8). The third number is the channel to be used. There are 16 disk channels, numbered 0-15. Channels 0 and 1 are used for loading and saving, and 15 is the command channel, so that leaves channels 2-14 for sequential files. It doesn't matter which channel you use, as long as it's not being used by another disk file. You can open more than one disk file, but each must have a different logical file and channel number.

The "S,W" after the filename means that the file will be sequential (S) and that you'll be writing (W) to it. Note the five commas in line 50; they're all necessary to separate the various parts of the OPEN command.

When the file is open, the red light on the front of the 1541 (or green light on the front of the 1571) drive will turn on and stay on until the file is closed. In line 60, PRINT# writes information to the file. It must be followed by

the logical file number, a comma, and the information. If line 5060 had been OPEN 5,8,3, line 60 would have used PRINT#5 instead of PRINT#1. Line 70 closes the file. CLOSE is followed by the logical file number.

Now that we've written a file called NAMES, here's a program to read the data.

10 OPEN 5,8,4,"NAMES,S,R"
20 INPUT#5,A\$,B\$,C\$
30 PRINT A\$:PRINT B\$:PRINT C\$
40 CLOSE 5

Since we're reading the file, there's an R, rather than a W, at the end of the OPEN command in line 10. In this instance, we're using logical file 5 and channel 4, although we could have used 1 and 2 as in the first program. INPUT# reads information from the file. Like PRINT#, it's followed by the logical file number and a comma. GET# acts like INPUT#, but it reads a single character at a time.

The programs have similar structures: They both INPUT from one source and PRINT to another. The first used INPUT/PRINT# to read the keyboard and write to a file, while the second used INPUT#/PRINT to read from the file and write to the screen.

Double-Width Printing

I use SpeedScript with my Star NX-1000C printer, but the PRINT command for double-width characters does not work. Is there a way to modify the program to use these commands, or should I use a Ctrl-£ command?

DON SYWASSINK
 SIERRA VISTA, AZ

A Ctrl-£, or stage 2, command should do the trick. With SpeedScript, you can define printkeys that will print whatever codes your printer uses for features such as double-width

or emphasized mode.

To define a printkey, at the top of your document press Ctrl-£ (or Ctrl-3), followed by the key that you want to assign as the printkey. Then enter the equal sign (=) and the ASCII value to be substituted for the printkey during printing. Many systems use an escape (ESC) code to break out of the word processor, and then certain ASCII values to activate various print modes.

For convenience, SpeedScript has already set four printkeys. Printkey 1 is defined as the escape key (ASCII 27). (With some printers and interfaces, you must send two escape codes to bypass the emulation.) Printkey 2 has a default value of 14, which is the ASCII code that puts most printers into double-width mode. Therefore, to switch to double-width mode, press Ctrl-£ and then press 1, press Ctrl-£ again, and then press 2. Next, enter the text you want printed in double-width mode.

Printkey 3 has a default value of 15, which turns off double-width on some printers and selects condensed mode on others. Printkey 4 is defined as 18, which selects reverse field on Commodore printers and some interfaces in emulation mode. On other printers, it switches to condensed mode. (See your printer manual for exact codes.)

To print the word WIDE in double width in the following example and then revert back to normal printing, your screen should look like this.

This is **12WIDE3** printing.

Remember, some printers require two escape codes. In that case, you would have **112** in front of the word WIDE. Codes can vary from printer to printer, so check your manual for specific values. □

How to create
 and use sequential
 files and use
 double-width printing
 with SpeedScript

MACHINE LANGUAGE

Jim Butterfield

CODING CHOICES

Recently, I saw the following message posted on a computer network: "I have a value in a single byte, and I want to calculate the remainder after dividing by 5. What code do you suggest?"

The remainder after division is often called the modulo; I don't know why the user wanted to calculate this, but there are several methods available that we can try. In this column, we'll discuss a couple of methods for solving the problem, and we'll also demonstrate the tradeoff between a program's speed and size. While we're at it, this might be a good time to gain some insight into hexadecimal numbers.

The standard method for solving this problem would be to use a conventional division routine that would yield both quotient and remainder. There are methods, however, that are designed either to achieve maximum speed or to utilize minimum memory. One rarely finds a piece of code that offers both. Almost all coding is a tradeoff between these two extremes.

A sample program called MOD5, printed at the end of this column, provides us with three approaches. The first routine offers speed, the second efficiency, and the third is a compromise of the two. You may want to examine the code of each one.

The fastest method is to look up the remainder in a table. Since a one-byte number can contain only 256 possible values, we can do this with a table of 256 bytes. This method couldn't be faster. We put the original byte into the Y register, and do the translation with a single instruction: LDA TABLE,Y. You'll find this at hex address 2015 in the program at the end of this column.

The method wastes memo-

ry, since we must devote 256 bytes to hold the table. The table could be loaded in, but it's quicker to calculate it when the program starts. You'll see this one-shot table build at addresses \$2000-\$2011. If only a few values were to be calculated, we couldn't justify this extra work. On the other hand, if there were thousands of values, this program would be speed efficient.

If the byte in question contains a value of 5 or more, we could subtract 5 and then repeat. Eventually, we end up with a value of 0 to 4; that's the remainder. The calculation loop, at addresses \$202C-\$2033, requires only four instructions: compare to 5, branch out if less (BCC), subtract 5, and branch back to the loop (BCS). Serious students of code will be able to explain why we don't need to set the C (carry) flag before subtraction and why the BCS (Branch Carry Set) command always branches.

The code is compact, fitting within eight bytes, but it could be slow. Since the original value could be as high as 255, the loop might be repeated as many as 51 times!

Most programs trade off speed against size. Programs that need to be fast will unfold their loops; this saves time but calls for more instructions. In this case, it really doesn't matter much. We have plenty of memory, and even the slowest method runs plenty fast for our purposes.

I wanted to add one more method, however. This third piece of code is moderately compact and fast. More important, it helps to show an interesting aspect of hex numbers.

It takes only a glance at a decimal number to tell whether it divides evenly by 5 or what the remainder would be. The last digit of the number tells the story (5 is a factor of

10, the base of decimal numbers). That's not true of hexadecimal numbers. The last digit will signal whether the number is divisible by 2, 4, 8 or 16, but it won't help you on the mod-5 question. Hex numbers such as 20 and 65 seem as if they should divide by 5, but they don't. Their decimal values are 32 and 101.

There is, however, a quick way to inspect hex numbers to see whether or not they will divide by 5. It's similar to the method we use with decimal numbers in testing whether or not a number divides by 9 or by 3. Add the decimal digits together; the total will have the same mod-9 value as the original number. Thus, decimal value 1234 will have a remainder of 1 when divided by 9. Calculate 1+2+3+4, giving 10, and the answer is a snap. The same holds true for division by 3, which is a factor of 9.

In hex, the sum of digits tells us about division by 15 or either of its factors (3 or 5). So, hex 23 will divide exactly by 5, and hex BC would have a remainder of 3. We know this because 2+3 gives 5, B+C or 11+12 gives 23, which would leave a remainder of 3 when divided by 5.

How would we do this in a computer program? A hex digit corresponds to four bits. We can extract the value of the high hex digit by shifting the number right four places. We extract the low digit value with a simple AND #\$0F. Add them together, and we have the sum of the two hex digits within a byte.

This sum cannot be greater than 30 (decimal), so we know that the simple subtraction of method 2 will now loop not more than six times. Quite an improvement from a possible 51 times around the loop.

Four LSR (Logical Shift Right) commands extract our high hex digit. We store the re-

When programming, there's usually the fastest way or the most compact way. Here's an attractive compromise.

sult and then call back the original value; masking with AND #50F isolates the low digit. Add them together (don't forget to clear the carry flag first with CLC), and we can repeat the subtract loop of method 2. The whole thing goes from hex address 2040 to 205B. That's a bit longer than the previous method, but there's quite a speed advantage.

The program works on almost any Commodore 8-bit computer. It first pokes the machine language code into place. Then it does the mod-5 calculation four times.

The first calculation is in BASIC, followed by each of the three above methods. The values used for the calculation are from ROM, hex addresses E000 through E006. You'll get the same results each time, of course.

You might want to use a machine language monitor to inspect the MOD5 code more closely. That'll give you an even better understanding of what's happening in the different routines.

```

100 DATA 162,0,160,0,152,157,
    0,33,200,192,5,144,2,160,0
110 DATA 232,208,242,188,0,224,
    185,0,33,9,48,32,210,255
120 DATA 232,224,7,144,240,169,
    13,76,210,255
130 DATA 162,0,189,0,224,201,5,
    144,4,233,5,176,248,9,48
140 DATA 32,210,255,232,224,7,
    144,235,176,226
150 DATA 162,0,189,0,224,72,74,
    74,74,74,141,255,32,104
160 DATA 41,15,24,109,255,32,
    201,5,144,4,233,5,176,248
170 DATA 9,48,32,210,255,232,
    224,7,144,220,176,186
200 FOR J=8192 TO 8295
210 READ X:T=T+X
220 POKE J,X
230 NEXT J
240 IF T<>12902 THEN STOP
400 PRINT "BASIC:"
410 FOR J=57344 TO 57350
420 X=PEEK(J):PRINT X-5*INT(X/5);
430 NEXT J
440 PRINT
450 PRINT "TABLE LOOKUP:"
460 SYS 8256
470 PRINT "SUBTRACT LOOP:"
480 SYS 8231
490 PRINT "HEX CHECKSUM:"
500 SYS 8256
510 PRINT "END."
```

C64/128 PUBLIC DOMAIN SOFTWARE

REQUEST FREE CATALOG or send \$2 for sample disk and catalog (REFUNDABLE). Categories include education, utilities, games, business, PRINT SHOP graphics, pre-tested programs and more. Rent for 75¢ or buy as low as \$1.00 per disk side or for 80¢ for 70 or more. \$20 order gets 4 free disks of your choice.

NEXT DAY SHIPPING!

SINCE 1986



CALOKE INDUSTRIES (Dept. GK)
PO BOX 18477, RAYTOWN, MO 64133



Circle Reader Service Number 181

Commodore Accessories & Necessities

Ribbons!
MPS 801 - 802 - 803 - 1525 - 1526 - 1000 -
1200 - 1230 - 1250
Commodore Printer (Comm. & PC Comp.)
C 64 C Computer • 1541-II Disk Drive
1802 Monitor

M3 Mouse
Modems
Joysticks
Icon Controller
Cables
Power Supply

Visa
MC or
UPS
COD

SOFTWARE: Educational - Productive - Fun • Commodore-Amiga Authorized
Dealer & Service Center • 24-Hour Turnaround on Repairs • CALL for PRICING



ELECTRO-TECH ELECTRONICS

677 East Main Street • Ventura, CA 93001 • 805-648-5417



Circle Reader Service Number 148

Calc II

Calc II makes your math work a breeze - whether it's a mortgage calculation, of data • Uppercase, lowercase and Commodore graphics all available • Bar

Same Old Ad - Great New Price!

Now get Calc II, the leading C64 spreadsheet, for the special year-end price of \$24.95, S&H included! The best now costs less - so order now, while the price is right!

US, CAN: \$24.95 (\$US/\$CDN), check/money order

OVERSEAS: \$24.95 US, International Money Order

decimal places, width and positioning | 7 weeks for delivery

PANKHURST PROGRAMMING Dept.G • Box 49135 • Montreal • Quebec • Canada • H1N 3T6

Circle Reader Service Number 152



NUCLEAR SUB COMMAND

Realistic Nuclear Attack Sub Simulation

C64 or 128 in 64 Mode
Command Missions Under The Arctic Ice
Hunt Russian Typhoons In The North Sea

Requires C64 GEOS 1.3 or 2.0

\$19.95 Check or Money Order

VMC Software PO Box 326

Cambria Hls. NY 11411



Circle Reader Service Number 171

Upgrade your Commodore system

Refurbished Hardware

MONITORS	DRIVES	OTHER
1701 - \$235	1541 - \$100	1660 - \$30
1702 - \$255	1541-II - \$120	1670 - \$50
1801 - \$265	1571 - \$165	C64 - \$100
1802 - \$285	1571-II - \$185	64C - \$120
1901 - \$295	1581 - \$180	C128 - \$175
1902 - \$305	1001SFD - \$150	C128D - \$225
10848 - \$325	1530 DATASETTE - \$35	

MANY BOOKS - \$10 SOFTWARE - \$10-20

ASK FOR ANYTHING, I MIGHT HAVE IT!

J.P. PBM PRODUCTS BY MAIL
P.O. BOX # 1233, STATION B
WESTON, ONTARIO, M9L2R9

New APROTEK modems

64/128/AMIGA-2400 BAUD - \$119
64/128/AMIGA-1200 BAUD - \$89
APROSAND-4 SLOT CARTRIDGE
EXPANDER FOR THE 64/128 - \$40

New CMD accessories

JIFFYDOS 64/128 & ANYDRIVE "SYSTEM" - \$85
128D/ANY DRIVE "SYSTEM" - \$95
ADDITIONAL DRIVE ROMS - \$45

RAMLINK/RAMCARD C/W BATTERY (0Mb) - \$345

1Mb RAM SIMM - \$75 4Mb RAM SIMM - \$250

SHIPPING INCLUDED FOR CANADA, USA & 15
15 DAY WARRANTY ON REFURBISHED GOODS
TAX - Canada + 7% GST, Ontario Res. + 8% PST

BEGINNER BASIC

Larry Cotton

ADDING ZIP TO BASIC

I get lots of requests for programming tips on ways to use BASIC with many applications, ranging from games to databases. A typical question might be, "How do I write a fast subroutine for doing searches for a given name and address in BASIC?" Another might be, "How do I make the aliens move faster while monitoring the joystick port, keeping score, and moving background scenery?"

When your BASIC programs need a burst of speed, give them a shot of machine language.

The answer to these questions is simple: If you want to do it fast, forget BASIC. Any program can be written in BASIC (assuming it will fit the computer's memory), but you might drop off to sleep waiting for something to happen.

Many articles have been written on maximizing BASIC's speed, and you can get some improvement using these techniques. However, none but the shortest, most elementary database programs should be written in BASIC. Any program that is more sophisticated is best written in some other programming language—preferably machine language (ML). To learn more about those programming techniques, consult Jim Butterfield's "Machine Language" column elsewhere in Gazette.

As for games, some can easily be written in pure BASIC, especially those that don't require blinding speed. Some examples would be word-search, spelling, math-drill, and even simulated board games. These types of games don't require much speed, and the user wouldn't notice if the computer slowed a little during execution.

Actually, BASIC and ML can be used together. One way is to use a BASIC program as an ML loader. Then a

SYS command puts you into machine language to stay.

The other way is to incorporate a speedy routine within a relatively slow BASIC program. Here's an example of the latter.

Suppose you're writing a pick-a-card-any-card game. You need to shuffle a deck of 52 cards quickly. By generating a nonrepeating list of 52 numbers, you could assign the numbers to an array of all the cards. The following program is one way to generate those numbers in BASIC.

BASIC RND

```
10 PRINT"[CLR][DOWN]PRESS  
ANY KEY TO RANDOMIZE 52  
NUMBERS  
20 PRINT"[DOWN] WITH OUT  
REPEATS.  
30 GETA$:IFA$="" THEN 30  
40 PRINTCHR$(147)  
50 Q=RND(-TI/101)  
60 C=52:DIMRN(C)  
70 FORX=1TOC  
80 RN(X)=INT (C*RND(1))+1  
90 FORT=XTO1STEP-1:  
IFRN(X)=RN(T-1) THEN80  
100 NEXT  
110 PRINTRN(X),  
120 NEXT  
130 PRINT"[DOWN] I'M SURE  
YOU DON'T WANT A REPEAT!
```

Now, let's try doing the same thing using machine language. (Don't worry, Jim Butterfield. Your column is safe!)

ML RND

```
10 Q=RND(-TI/101):  
PRINTCHR$(147)  
20 FORT=49152T049221:  
READD:POKET,D: NEXT  
30 POKE54286,255:  
POKE54287,255: POKE54290,  
128: REM SET UP VOICE 3  
40 CB=49480  
50 A=52:REM RANDOMIZES  
FROM 1 TO A WITHOUT  
REPEATS; MAX. VALUE OF A  
IS 255.  
60 POKE49222,A
```

```
70 PRINT"[DOWN] PRESS ANY  
KEY TO RANDOM-  
IZE"A"NUMBERS  
80 PRINT"[DOWN] WITH OUT  
REPEATS.  
90 GETA$:IFA$="" THEN90  
100 PRINTCHR$(147): SYS49152  
110 FORT=CB+1TOCB+A:  
PRINT(PEEK(T)),: NEXT  
120 PRINT:PRINT:PRINT"AGAIN?  
(Y=YES, N=NO)  
130 GETA$: IFA$<>"Y" THENIFA  
$<>"N" THEN130  
140 IFA$="N" THENEND  
150 GOTO100  
1000 DATA 172,70,192, 69,0,153,  
72,193,136,208,250, 173,  
70,192,170,160, 0,153,72  
1010 DATA 192,200,240,11,202,  
138,208,246,173,70, 192,  
170,76,17,192,173,70,192,  
141  
1020 DATA 71,192,173,27,212,  
170,189,72,192,172,70,  
192,217,72,193,240,241,  
136,208  
1030 DATA 248,172,71,192,153,  
72,193,206,71,192,208,  
227,96
```

Run both programs and observe the difference in how long it takes to generate 52 nonrepeating numbers. Allow plenty of time in the BASIC version, especially for the last several numbers.

To use embedded ML subroutines in a BASIC program, just SYS to the routine (see line 100 in ML RND). After the numbers are generated, they appear in memory registers 49481 through 49532 (for 52 numbers).

Here's an invitation to you programmers. I'd like to see your own versions of both BASIC and ML no-repeat randomizing programs. Please send them to me in care of COMPUTE's Gazette, 324 West Wendover Avenue, Suite 200, Greensboro, North Carolina 27408. If you keep them small enough to print on one page of the magazine, I'll publish the best examples in a future column. □

D'IVERSIONS

Fred D'Ignazio

CAPTAIN FUTURE AND HIS POCKET COMMANDER

Hello. This is Captain Future. People used to call me Fred, but that's when I was stationary, physical, and sitting in a real chair in a real office with real wires tying me to one spot.

Now I'm Captain Future. I'm mobile. I'm cordless. I'm wireless. I'm on the go. Where I call you from one minute is not where I'll be the next. You may be *there* (where you really are), but I'm only *here* in a metaphorical sense. I'm totally virtual. I beam you up from my little pocket phone somewhere on the planet. You beam me up, and I might be on a rock cliff or in my minivan or under a giant sequoia.

The revolution in my personal communications style occurred two months ago when I began renting my little Fujitsu Pocket Commander in a cellular phone. The phone weighs just a few ounces; it's about five inches long and two inches deep. I wear it in a little case on my belt.

When I'm wearing my Pocket Commander, I feel like a new man. With that little phone strapped to my side, I pretend I'm James Bond with his shoulder holster. But instead of a warlike secret agent, I'm a peaceful agent, armed for the future, ready to communicate with the world.

As soon as the Fujitsu lady checked me out on my new phone, I placed my very first call to my wife. I found her in an unlikely spot: the kitchen. She picked up the kitchen phone and said, "Hello?"

"Hello, dear," I said. "It's Captain Future, your husband."

"Where are you?" asked my wife, not at all impressed with my new secret identity.

"Right outside the kitchen

door, dear," I answered proudly. "About five feet away from you, in the driveway."

Next I called my mom. "Hello, Mom," I said. "It's your son, Captain Future."

"Who is this really?" my mother asked suspiciously.

"Aw, Mom," I said. "I'm calling you with no wires. No cables. Just thin air. And we're talking just like on a real phone. Isn't it grand?"

"I don't know any Captain Future," my mother said. "And whoever this is, you sound like you're calling me from inside a fish tank or a tin can. Please go away." Clink!

After calling my mom, I called everyone else I could think of. I called people from restaurants, bowling alleys, baseball diamonds, petting zoos, and public marinas.

Suddenly, I realized that I had become an addictive communicator. I first realized this after I installed the Fujitsu Pocket Commander in a cellular dock inside my minivan. Now I had a boosted power source, a cellular antenna corkscrewing up the side of my car, and an in-car speaker phone with a tiny mike clipped to the sun visor over the driver's seat. After I ran out of other people to call on my car phone, I began calling my wife again.

"Hello, dear!"

"Is that you, Fred?" my wife asked, from inside the house. "Where are you now?"

"Outside in the driveway, in our car."

"If you're already home, why don't you come inside and talk, like a real person?"

"Because it's more fun to call you from the car. It's kind of like an intercom. Besides, I've got my laptop computer out here, and I'm trying to plug it into the car phone so I can call online bulletin boards and maybe even send faxes."

"Why would you want to

send faxes from your car?" my wife asked. "Especially when you're parked in our driveway?"

Since then, my wife has slowly warmed to pocket phones. For example, last week she and I were trekking around a rock quarry on the seacoast north of Boston. There wasn't another person for miles around. Nature was in bloom all around us. Suddenly, my wife reached for my belt.

"Dear!" I screamed, jumping backward. "What's got into you?"

"Your phone," she said. "I want your phone. I just remembered I have to call my office."

While my wife sat on the quarry's edge talking with her boss and her secretary, I began climbing down the vertical wall of the quarry. After about 15 minutes, I made it down to the level of the water that filled the quarry's inner basin. I took off my shoes and dangled my bare toes in the water, scaring away a couple of polliwogs that were sunning themselves on a big boulder just beneath the surface. I listened to my wife as she talked on the cellular phone, her voice crystal clear high above.

"This is weird," I thought. Somehow, my wife's phone call to her office didn't seem out of place even here, deep in the heart of undisturbed nature. In addition, the call didn't stress me out or make me lose my sense of awe and appreciation for my surroundings. Somehow, everything seemed to fit in.

It'll be amazing to see how this revolution changes the future face of work and leisure. Maybe in the future it'll be normal to conduct business on a rock cliff while on a daylong trek into a remote granite quarry. As I gazed out at the deep blue quarry lake all around me, I thought that might be kind of nice. □

In this exciting episode, read how a mild-mannered magazine columnist is transformed into Captain Future.

Steve Vander Ark

ULTIMATE GEOS

In an IBM magazine recently, a senior editor describes his quest for the ultimate PC. The cost of this system would buy a pretty nice sports car.

That started me thinking about the ultimate GEOS setup. I wondered just how powerful GEOS could be with all the right gizmos hooked up to it. And, since Christmas is just about once again to take over prime time and the malls, I figure this is a great time to make yet another GEOS wish list. While the total wouldn't buy a snazzy sports car, it might be enough to buy, oh, a used Ford Escort.

My dream GEOS setup has to start with a computer, of course. I'll go with the 128, since an 80-column screen is essential. Now, the 128D does have a detachable keyboard, which is nice, and an extra 64K of video RAM, but I don't like the idea of having that darn 1571 permanently set up as drive 8. I have much better ideas for disk drives, so I'll stick with the flat 128.

One advantage to the Commodore computer is that you don't have to spend heaps of money on extra cards to do things like create color screen displays. Our 128 has 40-column and 80-column modes built right in; all we need to do is to choose a monitor which can display either mode on command. Since nothing but the best will do for our ultimate setup, I'll add a Commodore 1084S monitor.

Mode switching can become a constant chore when you work with GEOS on the 128; many programs, from little utilities like Blue Pencil to big utilities like geoPublish, run fine on the 128 but demand 40 columns. To make life a little easier, I'll add a 13-inch 40-column monitor on the side. You'd be surprised how

handy this configuration can be. When you switch to 40-column mode, the image jumps from one monitor to the other, and the screen of the unused monitor goes peacefully blank. If you can't afford a second monitor, a color TV works about as well. I'm going for broke here, though, so I'll pick up an 1802 monitor.

One or two more details are needed before we tackle the big question of drives and RAM expansion. We must, for example, have an input device. Speaking from experience, having used a joystick, mouse, KoalaPad, and light pen with an assortment of drivers, I strongly recommend a mouse. Speaking from the experience of friends, the mouse of choice is the Commodore 1351.

OK, let's talk disk drives. It would be nice to include drives to handle both 5¼-inch and 3½-inch disks. For the 5¼-inch disks, the best bet is the good old 1571, which can read single- or double-sided floppies. That's pretty much standard stuff.

Let's take a leap into the big leagues for the 3½-inch drive. We have a couple of very impressive choices, now that Creative Micro Designs (CMD) has released a pair of high-density drives: the FD-2000, with 1.6 megs per disk, and the FD-4000, with a whopping 3.2 megs of data on a floppy! We're talking dream material here, folks! The ultimate GEOS system has to have an FD-4000.

That accounts for two of the drives. GEOS can effectively handle only three drives, so this next choice might be a little sticky. Some form of RAM expansion is a must with GEOS, but if it's configured as a RAM drive, there goes the third drive. It's hard to imagine an ultimate system, however, without a hard drive. For now,

anyway, I'll just choose both.

The hard drive of choice will be one of the CMD HD-series drives, which are compatible with GEOS and practically everything else. Since money's no object, I'll take the HD-200 with 200MB capacity.

I do need RAM expansion as well, so let's take a look at the options. The Commodore 1751 RAM expansion unit can be upgraded to larger capacities than the stock 512K, but it's still a pretty bland unit. A much more exciting choice would be either the RAMLink or RAMDrive from CMD. Each has two invaluable features no RAM expansion device should be without: a separate power supply, which keeps the data intact when you shut down your system, and a battery backup, which means that in the event of a power failure, your data won't evaporate like spit on a hot skillet. Both are fine units. RAMLink can be upgraded to 16 megs, while RAMDrive is limited to 8 megs. RAMLink also can be fitted out with a realtime clock circuit to set your clock in GEOS, and it also features a pass-through port that I just might need before this system is completed. I'll add RAMLink, maxed out to 16 megs.

I'll have to decide how to configure all those drives when I pick a desktop program, but I'll do that next month when I talk about software. For now, let's recap my shopping list.

128 CPU (used)	\$ 200.00
1084S monitor	\$ 289.00
1802 monitor (reconditioned)	\$ 99.95
1351 mouse	\$ 32.95
1571 disk drive (used)	\$ 100.00
FD-4000 disk drive	\$ 300.00
HD-200 hard drive	\$ 1,099.95
RAMLink with battery and 16MB RAM	\$ 584.90
TOTAL	\$ 2,706.75 ☐

Now that the holiday season is fast approaching, here's the GEOS system I'd really like to find under the tree.

NEW PRODUCTS From Makers of RAMDRIVE



BBG RAM

Battery Back-up
Ram Disk for
GEOS 2.0 and
GEOS 128, 2.0

- Magnitudes faster than any floppy or hard drive
- 2 MEG model has capacity of TEN 1541's
- Includes GEOS application to select one of up to five 1571's
- Reboots GEOS from BBG Ram quickly and quietly
- Supplied with wall mount power supply and battery cable and holder
- Automatically detects power out and switches to back-up mode
- Activity light indicates access
- Battery used only when wall mount AC power supply off

INTRODUCTORY PRICE

MODEL 512

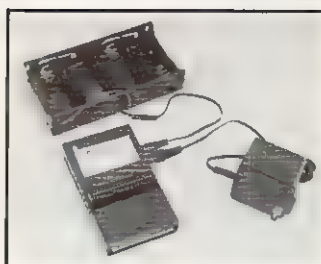
\$79⁰⁰

1 MEG

\$110⁰⁰

2 MEG

\$139⁰⁰



BBU

Battery Back-up
Interface
Module for
Commodore
17xx REU's and
Berkley Softworks'
GEORAM 512

- Reset button without data loss
- Activity indicator light during access
- Battery low voltage indicator
- Wall-mounted power supply and battery holder and cable supplied
- GEOS compatible, allows reboot to GEOS
- Automatic battery back-up, no switches to push
- Battery powers unit only when AC power off
- BBU supplies power to 17xx REU's and GEORAM. Commodore heavy power supply not required

INTRODUCTORY PRICE

\$49⁰⁰

Call: 1-800-925-9774

GEOS registered Trademark of Berkley Softworks, Inc.



PERFORMANCE
PERIPHERALS Inc.

5 Upper Loudon Road
Loudonville, New York 12211

Please Add:
U.S. \$6.00 S&H
Canada \$10.00 S&H
\$4.00 C.O.D.

Circle Reader Service Number 153

COMPUTE's SpeedScript Disk

A powerful word processing
package for Commodore 64
and 128 owners

A Great Deal for Commodore Users!

- SpeedScript for the 64
- SpeedScript 128—80-column version
- Spelling checkers
- Mail merge
- Date-and-time stamp
- 80-column preview for the 64
- Turbo save and load
- Plus more than a dozen other SpeedScript support utilities all on one disk (including full documentation)

**ONLY
\$11.95**

YES! Send me _____ copies of COMPUTE's
SpeedScript Disk.

I've enclosed \$11.95 plus \$2.00 postage and handling. (Outside
U.S. and Canada add \$1.00 for surface mail or \$3.00 for
airmail.)

ORDER NOW!

_____ Amount
_____ Sales Tax*
_____ Total

Name _____

Address _____

City _____ State _____ ZIP _____

Mail personal check or money order to

Commodore SpeedScript Disk
324 W. Wendover Ave., Ste. 200
Greensboro, NC 27408

Residents of North Carolina and New York, add appropriate tax for your area. Canadian
orders, add 7% good and services tax

Please allow 4-6 weeks for delivery. Program available only on 5¼-inch disks.

PROGRAMMER'S PAGE

Randy Thompson

READER'S GRAB BAG

From the mailbox to the printer, this grab bag of programming gems comes from you, our readers. Keep 'em coming. We pay up to \$50 for each tip we publish.

64 or 128?

There's an easy way for your BASIC program to detect which 8-bit Commodore computer it's running on. Simply check the variable DS\$ in the first line of your program. If DS\$ is equal to a null string (DS\$=""), your program is running on a 64 or on a 128 operating in 64 mode. That's because in 128 mode, the DS\$ string returns the current status of the disk drive, whereas on a 64, DS\$ doesn't hold anything until you define it.

Incidentally, checking DS\$ on a 128 that has no drive attached can crash your program, but how many driveless 128 owners do you know?)

ARTHUR MOORE
ORLANDO, FL

Redefining Restore

This two-line program turns your Restore key into a computer reset button. After you type in and run the program, tapping (sharply, of course) the Restore key will yield the same results as the BASIC command SYS 64738.

```
10 FOR I=32768 TO 32776:  
  READD: POKEI,D: NEXT  
20 DATA 248,252,226,252,195,  
  194,205,56,48
```

To disable your new reset key, turn the computer off and then on again.

Here's how the program works. Whenever you press the Restore key, the computer checks memory locations 32772-32776 for the numbers 195, 194, 205, 56, and 48.

These numbers are the PETSCII codes for the reversed capital letters *CBM* followed by the number 80. If that string is found, the computer jumps to the machine language subroutine pointed to by memory locations 32770 and 32771. The program listed above redirects this vector to point to the 64's reset routine found at 64738. Things get a bit tricky here, because the reset routine at 64738 also looks at memory locations 32772-32776 for the string *CBM80*. If it finds those characters, it jumps to the subroutine pointed to by the vector at 32768. To avoid such jumpy behavior, our Restore-reset routine sets this vector so that it points right back into the 64's reset routine, forcing the computer to continue the reset operation from where it left off.

One of the neat features of this program is that you can set the vector found at 32770 so that it points to your own machine language program—one that will execute every time you press Restore. In the program above, this vector is set equal to the third and fourth numbers found in the DATA statement on line 20.

Note that this program disrupts the normal operation of the Run/Stop-Restore key combination. Now, pressing Run/Stop-Restore resets the computer, also, but it clears any program that may have been in memory.

LANCE SLOAN
SWARTZ CREEK, MI

Convenient Comma Key

This hack is for 128 owners who enter a lot of data via their numeric keypads. It transforms the keypad's Enter key into a comma key. Such a set-up is ideal for people who type in a lot of MLX listings.

```
10 FOR I=0 TO 28: READ D:
```

```
POKE 4864+I,D:C=C+D: NEXT  
20 IF C<>3231 THEN PRINT  
  "ERROR IN DATA STATE  
  MENTS": END  
30 BANK 15: SYS 4864: PRINT  
  "NUMERIC COMMA KEY  
  ACTIVE"  
40 PRINT "TO DISABLE: POKE  
  830,128:POKE 831,250"  
50 PRINT "TO REACTIVATE:  
  BANK 15:SYS 4864"  
60 DATA 160,0,185,128,250,  
  153,29,19  
70 DATA 200,192,89,208,245,  
  169,19,141  
80 DATA 63,3,169,29,141,  
  62,3,169  
90 DATA 44,141,105,19,96
```

EMIL HEYROVSKY
PRAGUE, CZECHOSLOVAKIA

ReDIMing Arrays

If you ever want to erase and/or redimension (DIM) your variable arrays, execute the following two commands from within your program.

```
POKE 49,PEEK(47): POKE  
50,PEEK(48)
```

This will erase all arrays. Unlike the CLR command, however, these POKEs will not affect nonarray variables.

HELEN ROTH
LOS ANGELES, CA

Monitoring 64 Code on the 128

The most popular area for programmers to store machine language programs on the 64 is in the 4K area starting at 49152 (\$C000). Of course, this area is relatively useless on the 128 because 49152 is where editor ROM is mapped, but that doesn't mean you'd never want to load your 64 code here. Because RAM underlies 128 editor ROM, 64 machine code can be stored here and worked on using the 128's built-in machine language monitor.

YANNICK TROTTIER
BRIDGETOWN, NS
CANADA

Readers take over
this month's column
with a collection
of handy tips for the
64 and 128.

PROGRAMS

MOB MASTER

By Hong Pham

Sprites (or movable object blocks) are large user-defined graphics that can be placed anywhere on your monitor's screen. The 128 has a powerful sprite controller that is built into its BASIC operating system to make sprite programming fairly easy. The 64, which has the same sprite capabilities as the 128, lacks the 128's sprite controller system. Programming sprites on a 64 usually requires many lines of code filled with awkward POKes—but now there's MOB Master.

MOB Master gives the 64 many of the same features and sprite commands that are found on a 128. It also has extras, such as sprite animation and boundary-handling commands.

While this article explains how to use MOB Master's commands, it doesn't provide a complete tutorial for creating and using sprites. Programmers who already use sprites should have no trouble using MOB Master. Beginners can find more detailed descriptions of sprites and video banking in such reference books as *Commodore 64 Programmer's Reference Guide* or *Mapping the Commodore 64*.

Getting Started

MOB Master is written entirely in machine language. Use MLX, our machine language entry program, to type it in. If you don't have a copy of MLX, see "Typing Aids" elsewhere in this section. When MLX prompts, respond with the following values.

Starting address: 7D00

Ending address: 86EF

When you've finished typing in MOB Master, be sure to save it before exiting MLX.

To activate MOB Master, load it with the .8,1 extension and then type *SYS 32000*. At this point you'll see a title screen that lets you know MOB Master has been activated. You may now begin writing your own sprite program. Instead of using cumbersome POKes to control your sprites, however, you'll have a whole new library of commands at your disposal.

Ten Sprite Commands

MOB Master adds ten new BASIC com-

mands for easier sprite definition, positioning, movement, animation, and other miscellaneous functions. The first three commands are similar to the 128's sprite commands of the same name.

Here's an important programming note to remember: When using a MOB Master command within a BASIC program, you must precede that command with a slash (/). In immediate mode, however, you don't need to use the slash.

SPRITE #, on/off, fgnd, priority, x-exp, y-exp, mode

The SPRITE command defines most of the characteristics of a sprite. Select the sprite number (#) with a value ranging from 0 to 7.

Use a 1 in the *on/off* parameter to turn on your sprite; use a 0 to turn it off.

The sprite foreground (*fgnd*) color is defined with a value between 0 and 16.

To make the sprite appear in front of objects on the screen, set its *priority* parameter to 0. To make it appear behind the objects on the screen, set the parameter to 1.

The sprite can expand to twice its original size horizontally (*x-exp*) or vertically (*y-exp*) by setting the next two parameters to 1. Set these parameters to 0 to turn off sprite expansion.

Turn on multicolor *mode* with a 1 or turn it off with a 0.

MOVSPR #, x, y

MOVSPR either positions or moves the sprite. The first example plots the sprite anywhere on the screen, with *x* being any pixel number between 0 and 319 and *y* any number between 0 and 199. Unlike normal sprite programming, MOB Master lets you place sprites beyond the 255th pixel without additional programming.

MOVSPR #, direction # speed

This variation moves the sprite in a specific direction and speed. The *direction* value can range from 0 to 255. This value can be converted to degrees by multiplying it by 45/32. To move the sprite up, use a value of 0. To move it to the

right, use a value of 64. To move down, use 128. To move left, use 192. Intermediate values will move the sprite at different angles across the screen.

The value for *speed* can range from 0 to 255, with 0 being fastest and 254 being slowest. A value of 255 means that the sprite is stationary.

The format for this command is similar to that used for positioning a sprite, but instead of using a comma to separate the values, use the # sign. For example, *MOVSPR 0, 64 # 100* would move sprite 0 to the right at a fairly slow speed.

SPRCOLOR color 1, color 2

In multicolor mode, the two multicolor colors are shared among all eight sprites. The first parameter defines multicolor 0, and the second parameter defines multicolor 1.

ANIMATE #, speed, mode, start frame, end frame

ANIMATE defines a sprite image or animates the sprite by successively changing its image pointers. The animation *speed* can range from 0 to 255, with 0 being fastest and 254 being slowest. A value of 255 means that the sprite has no animation. The *mode* parameter tells MOB Master how the sprite will be animated. A value of 0 means that the sprite will always be animated, and a 1 means that the sprite will be animated just once. Any other value will stop the sprite from being animated.

Sprite data resides in blocks of 64 bytes each. These blocks are numbered from 0-255. To calculate the location of a block of sprite data in memory, multiply the block number by 64. The result gives you the location where the first byte of a sprite definition should be poked. If you define several sprites whose shapes differ slightly and then switch rapidly among these blocks with the ANIMATE command, the sprite will appear to move in an animated fashion.

The *start frame* parameter indicates the first sprite image or block for animation. The *end frame* parameter indicates the ending block number for the animated sequence. Any sprite

frames that are in between these will be automatically called.

BOUNDARY #, mode, top, bottom, left, right

Each sprite has its own individual screen boundaries. Once the sprite reaches a boundary that you set with a pixel number, the *mode* parameter indicates the action that the sprite will take. A 0 means that the sprite will wrap around and be placed on the opposite boundary. A 1 indicates that the sprite will bounce off the boundary. A 2 indicates that the sprite will stop at the boundary. Any other value indicates that the sprite will be turned off when it reaches a boundary, discontinuing motion.

For convenience, MOB Master allows only the horizontal boundary to be accurate to within two pixels. The actual boundary occurs on every even pixel. MOB Master will automatically divide the value that you have supplied with the boundary parameter by 2.

BOUNCE #, mode

BOUNCE bounces a sprite in a certain way, even if it's not at its boundary. *Mode* indicates how the sprite will bounce. A 0 argument means that the sprite will bounce vertically; a 1 indicates that the sprite will bounce laterally. Any other value will reverse the sprite's direction.

SPLIT mode

MOB Master supports two different raster-interrupt routines for flicker-free sprites. It accomplishes this task by updating its shadow registers when the raster scan is at a certain position on the screen. To select one of the two raster-interrupt routines, set *mode* to either 0 or 1. The only difference is that the latter routine allows you to display sprites on the top and bottom borders. If no argument follows SPLIT, it will turn off the raster-interrupt routine.

Before attempting an input or output operation, especially with a disk drive, it's best that you turn off the raster-interrupt routine. If you don't turn off the routine, the computer may freeze until you hit the Run/Stop and Restore keys simultaneously.

IRQ enable/disable

When you move multiple sprites as if they were one sprite, one sprite may move ahead of the others, creating a gap. This is because MOB Master updates the sprites 60 times a second, and BASIC may be too slow to move all the sprites before MOB Master updates them. One sprite may be updated before BASIC updates the others.

To temporarily stop MOB Master from updating the sprites, use IRQ 0. Any other value will allow MOB Master to continue updating the sprites. Be careful not to hold the system for too long, or the computer may hang up.

ZAP

ZAP clears all the sprite registers.

KILL

KILL disables MOB Master and restores the previous interrupt and BASIC vectors.

Additional Notes

For all MOB Master statements, with the exception of IRQ, you can substitute an unknown parameter with an asterisk (*). You can also use the asterisk if you don't want to make any changes to the current parameter. You don't have to supply all of the parameters of the command, but you must denote the sprite number. You cannot substitute an asterisk for the sprite number.

Collision Detection

Sprite-to-sprite or sprite-to-background collisions can be monitored by using the USR command. To return the status of the last sprite-to-sprite collision, type in *PRINT USR(0)*. Likewise, to return the status of the last sprite-to-background status, type *PRINT USR(1)*.

Shadow Registers

MOB Master updates its shadow registers to the VIC-II during a raster interrupt, or once every 1/60 of a second, to avoid sprite flickers. An advantage of this setup is that the sprites continue to move while your program does something else. You shouldn't make a direct POKE to the VIC-II registers to define a sprite, because once a raster inter-

rupt occurs, MOB Master overwrites the VIC-II register with the contents of the shadow register. Therefore, poke to the shadow register instead. Below is the memory map of the shadow register and its VIC-II equivalent.

VIC-II Location	Equivalent Shadow Register	Description
	(Base + offset)	
\$D000 (53248)	Base + 1312	Sprite 0 x position
\$D001 (53249)	Base + 1320	Sprite 0 y position
\$D010 (53264)	Base + 1328	Most significant bits of sprites 0-7 horizontal positions
\$D015 (53269)	Base + 1329	Sprite enable register
\$D017 (53271)	Base + 1330	Sprite Y-Expand register
\$D01D (53277)	Base + 1331	Sprite X-Expand register
\$D01B (53275)	Base + 1332	Sprite-to-foreground priority register
\$D01C (53276)	Base + 1333	Sprite multi-color mode register
\$D025 (53285)	Base + 1334	Sprite multi-color register 0
\$D026 (53286)	Base + 1335	Sprite multi-color register 1
\$D027 (53287)	Base + 1336	Sprite 0 color register
\$07F8 (2040)	Base + 1344	Sprite shape data pointers. The actual location of this register depends on the location of the video matrix.

The default base is \$7D00 (32000).

MOB Master and Machine Language

MOB Master's sprite-handling ability is not restricted to BASIC programs. Machine language programmers will find

MOB Master useful, as well. In fact, MOB Master and machine language are a great combination because you can do much more with machine language than you can with BASIC.

To make access to MOB Master's subroutines easier, MOB Master has a jump table. For all of MOB Master's subroutines, enter it with a JSR instruction, and use the X register to denote the sprite number. The following is the memory layout of the jump table.

Location	Description
(Base + offset)	
Base	Enable MOB Master's BASIC interface.
Base + 3	Enable raster-interrupt routine 1.
Base + 6	Enable raster-interrupt routine 2.
Base + 9	Disable raster-interrupt routine.
Base + 12	Zap all sprite registers.
Base + 15	Turn sprite on or off; C flag set = sprite is on.
Base + 18	Position sprite at x, y. AC = LSB of x position; C flag = MSB of x position; YR = y position.
Base + 21	Set sprite color; put sprite color in AC.
Base + 24	Set sprite multicolor mode characteristics. C flag set = multicolor mode on. AC = multicolor 0; YR = multicolor 1.
Base + 27	Set sprite to background priority; C flag set = background has priority.
Base + 30	Set Y-expand; C flag set = expand sprite vertically.
Base + 33	Set X-expand; C flag set = expand sprite horizontally.
Base + 36	Set sprite speed; AC = sprite speed.
Base + 39	Set boundary action mode (similar to BASIC BOUNDARY statement).
Base + 42	Set sprite direction;

Base + 45	AC = sprite direction. Set animation speed and mode. AC = animation speed; YR = mode.
Base + 48	Set animation start and end image pointers. AC = start image location; YR = end image location.
Base + 51	Set top and bottom borders. AC = top border; YR = bottom border.
Base + 54	Set left and right borders. AC = left border; YR = right border. Note: Divide border value by 2.
Base + 57	Bounce sprite vertically.
Base + 60	Bounce sprite laterally.
Base + 63	Reverse sprite direction.

Note: C flag = Carry flag, AC = Accumulator, XR = X register, YR = Y register

If you're using MOB Master exclusively in machine language, you may delete the BASIC interface module starting at location \$82CC (33484) or (base) + 1484 to \$86EA 34538 or (base) + 2538.

MOB MASTER

7D00:4C	CC 82 4C 7A 7D 4C 81 4D
7D08:7D	4C 88 7D 4C A1 7D 4C ED
7D10:AB	81 4C 94 81 4C BC 81 4C
7D18:4C	C2 81 4C D9 81 4C EA 37
7D20:81	4C FB 81 4C 60 81 4C B9
7D28:80	81 4C 69 81 4C 6D 81 7D
7D30:4C	75 81 4C 84 81 4C 8C F2
7D38:81	4C 12 81 4C 1F 81 4C 8F
7D40:29	81 78 CD 14 03 D0 05 6F
7D48:EC	15 03 F0 0E 48 AD 14 6F
7D50:03	8D DA 7D AD 15 03 8D B8
7D58:DB	7D 68 8D 14 03 8E 15 65
7D60:03	A9 7F 8D 0D DC A9 81 C0
7D68:8D	1A D0 AD 11 D0 29 7F 43
7D70:8D	11 D0 AD 1E D0 AD 1F 1A
7D78:D0	60 A9 B9 A2 7D 4C 42 AA
7D80:7D	A9 DC A2 7D 4C 42 7D 89
7D88:78	A9 81 8D 0D DC A9 00 62
7D90:8D	1A D0 AD DA 7D 8D 14 C9
7D98:03	AD DB 7D 8D 15 03 58 F2
7DA0:60	A2 80 A9 00 9D 20 82 58
7DA8:CA	10 FA A2 07 A9 FF 9D 13
7DB0:48	82 9D 78 82 CA 10 F7 03
7DB8:60	AD 19 D0 8D 19 D0 29 1B
7DC0:01	F0 16 20 1E 7E AD 11 94
7DC8:D0	29 7F 8D 11 D0 A9 FA 59

7DD0:8D	12 D0 20 BC 7F 20 70 C7
7DD8:7E	4C 31 EA AD 19 D0 8D FB
7DE0:19	D0 29 01 F0 19 A9 00 11
7DE8:D0	21 20 1E 7E AD 11 D0 18
7DF0:29	7F 09 08 8D 11 D0 A9 FD
7DF8:F9	8D 12 D0 EE E7 7D 20 D5
7E00:BC	7F 20 70 7E 68 A8 68 95
7E08:AA	68 40 AD 11 D0 29 77 EC
7E10:8D	11 D0 A9 00 8D E7 7D 50
7E18:8D	12 D0 4C D9 7D A2 07 50
7E20:A0	0E BD 20 82 99 00 D0 F5
7E28:BD	28 82 99 01 D0 BD 38 F6
7E30:82	9D 27 D0 BD 40 82 9D 59
7E38:F8	FF 88 88 CA 10 E3 AD 57
7E40:30	82 8D 10 D0 AD 31 82 CA
7E48:8D	15 D0 AD 32 82 8D 17 14
7E50:D0	AD 33 82 8D 1D D0 AD DF
7E58:34	82 8D 1B D0 AD 35 82 9D
7E60:8D	1C D0 AD 36 82 8D 25 1C
7E68:D0	AD 37 82 8D 26 D0 60 4F
7E70:A2	07 BD 48 82 C9 FF F0 E8
7E78:11	C9 40 90 14 BD 50 82 3C
7E80:F0	0F DE 50 82 D0 03 20 18
7E88:9D	7E 20 83 7F CA 10 E2 5A
7E90:60	38 BD 48 82 E9 3F 9D DF
7E98:50	82 4C 87 7E BD 58 82 7E
7EA0:DD	60 82 F0 09 9D 60 82 06
7EA8:20	2E 7F 9D 68 82 BD 58 2C
7EB0:82	C9 20 90 21 C9 40 90 AF
7EB8:26	C9 60 90 2B C9 80 90 62
7EC0:30	C9 A0 90 35 C9 C0 90 48
7EC8:3A	C9 E0 90 3F 20 45 7F FE
7ED0:20	15 7F 4C 4E 7F 20 45 CD
7ED8:7F	20 15 7F 4C 48 7F 20 DA
7EE0:48	7F 20 15 7F 4C 45 7F 6E
7EE8:20	48 7F 20 15 7F 4C 4B 84
7EF0:7F	20 4B 7F 20 15 7F 4C B7
7EF8:48	7F 20 4B 7F 20 15 7F D8
7F00:4C	4E 7F 20 4E 7F 20 15 70
7F08:7F	4C 4B 7F 20 4E 7F 20 94
7F10:15	7F 4C 45 7F 20 2E 7F AF
7F18:18	7D 68 82 C9 20 90 08 AF
7F20:38	E9 20 9D 68 82 38 60 B1
7F28:9D	68 82 68 68 60 BD 58 7F
7F30:82	29 20 8D BD 50 82 29 BC
7F38:1F	28 F0 08 8D 43 7F 38 20
7F40:A9	20 E9 FF 60 A0 00 2C 0B
7F48:A0	01 2C A0 02 2C A0 03 6C
7F50:BD	48 82 C9 40 B0 1E 38 66
7F58:A9	41 FD 48 82 20 D0 80 77
7F60:98	F0 09 88 F0 09 88 F0 3F
7F68:09	4C 98 80 4C D1 7F 4C 0F
7F70:49	80 4C 16 80 BD 48 82 2D
7F78:38	E9 3F 9D 50 82 A9 01 B0
7F80:4C	5D 7F BD 98 82 30 0F 07
7F88:BD	78 82 C9 FF F0 08 BD 03
7F90:80	82 F0 04 DE 80 82 60 2D
7F98:BD	78 82 9D 80 82 BD 40 88
7FA0:82	DD 90 82 F0 04 FE 40 68
7FA8:82	60 BD 98 82 F0 06 A9 CF
7FB0:FF	9D 98 82 60 BD 88 82 DF
7FB8:FD	40 82 60 AD 00 DD 29 3F
7FC0:03	AA AD 18 D0 29 F0 4A 7A
7FC8:4A	18 7D 1C 82 8D 39 7E 9F
7FD0:60	BD 28 82 DD A0 82 B0 C3
7FD8:33	20 5B 81 F0 27 88 F0 23
7FE0:1E	88 F0 0F AD 31 82 3D 94
7FE8:14	82 8D 31 82 A9 FF 9D AF
7FF0:48	82 60 BD A0 82 9D 28 0F
7FF8:82	A9 FF 9D 48 82 60 20 AA

PROGRAMS

```

8000:12 81 4C 3F 80 BD A8 82 B6
8008:9D 28 82 60 38 BD 28 82 C3
8010:89 01 9D 28 82 60 BD A8 36
8018:82 DD 28 82 B0 21 20 5B A4
8020:81 F0 15 88 F0 0C 88 F0 03
8028:03 4C E4 7F BD A8 82 4C 34
8030:F6 7F 20 12 81 4C 0C 80 87
8038:BD A0 82 9D 28 82 60 18 8E
8040:BD 28 82 69 01 9D 28 82 62
8048:60 20 DD 80 BD B8 82 20 3B
8050:55 81 F0 02 B0 31 20 5B 80
8058:81 F0 23 88 F0 1A 88 F0 35
8060:03 4C E4 7F A9 FF 9D 48 5B
8068:82 AD CA 82 8D C8 82 AD D9
8070:CB 82 8D C9 82 4C F4 80 F5
8078:20 1F 81 4C BF 00 BD B0 72
8080:82 20 32 81 4C 69 80 18 4A
8088:AD C8 82 69 01 8D C8 82 CB
8090:90 03 EE C9 82 4C F4 80 C4
8098:20 DD 80 BD B0 82 20 55 32
80A0:81 F0 02 90 1A 20 5B 81 71
80AB:F0 0F 88 F0 06 88 F0 B4 EE
80B0:4C E4 7F 20 1F 81 4C 87 22
80B8:80 BD B8 82 4C 81 80 38 4A
80C0:AD C8 82 E9 01 8D C8 82 0C
80C8:B0 03 CE C9 82 4C F4 80 09
80D0:8D 11 80 8D 44 80 8D 8C 91
80D8:80 8D C4 80 60 AD 30 82 BA
80E0:3D 0C 82 D0 03 A9 00 2C CB
80E8:A9 01 8D C9 82 BD 20 82 1B
80F0:8D C8 82 60 AD C8 82 9D 74
80F8:20 82 AD 30 82 3D 14 82 17
8100:8D 30 82 AD C9 82 F0 09 44
8108:AD 30 82 1D 0C 82 8D 30 C5
8110:82 60 38 A9 40 FD 58 82 3B
8118:18 69 40 9D 58 82 60 A9 9A
8120:00 38 FD 58 82 9D 58 82 34
8128:60 BD 58 82 49 80 9D 58 DD
8130:82 60 18 0A 8D CA 82 90 5D
8138:03 A9 01 2C A9 00 8D CB 3E
8140:82 60 38 AD CA 82 ED C8 83
8148:82 8D 53 81 AD CB 82 ED 02
8150:C9 82 09 FF 60 20 32 81 63
8158:4C 42 81 BD C0 82 A8 60 DF
8160:9D 48 82 A9 00 9D 50 82 C8
8168:60 9D 58 82 60 9D 78 82 23
8170:98 9D 98 82 60 9D 88 82 6F
8178:9D 40 82 98 9D 90 82 60 C8
8180:9D 70 82 60 9D A0 82 98 D1
8188:9D A8 82 60 9D B0 82 98 28
8190:9D B8 82 60 8D C8 82 B0 2C
8198:03 A9 00 2C A9 01 8D C9 80
81A0:82 78 20 F4 80 58 98 9D 8A
81A8:28 82 60 AD 31 82 90 05 01
81B0:1D 0C 82 B0 03 3D 14 82 58
81B8:8D 31 82 60 29 0F 9D 38 1E
81C0:82 60 48 AD 35 82 3D 14 43
81C8:82 90 03 1D 0C 82 8D 35 1E
81D0:82 68 8D 36 82 8C 37 82 7B
81D8:60 AD 34 82 90 05 1D 0C 05
81E0:82 B0 03 3D 14 82 8D 34 7F
81E8:82 60 AD 32 82 90 05 1D 9B
81F0:0C 82 B0 03 3D 14 82 8D AD
81F8:32 82 60 AD 33 82 90 05 66
8200:1D 0C 82 B0 03 3D 14 82 A9
8208:8D 33 82 60 01 02 04 08 17
8210:10 20 40 80 FE FD FB F7 15
8218:EF DF BF 7F C3 83 43 03 B2
8220:00 00 00 00 00 00 00 25
8228:00 00 00 00 00 00 00 2D
8230:00 00 00 00 00 00 00 35
8238:00 00 00 00 00 00 00 3D
8240:00 00 00 00 00 00 00 45
8248:FF FF FF FF FF FF FF 4D
8250:00 00 00 00 00 00 00 55
8258:00 00 00 00 00 00 00 5D
8260:00 00 00 00 00 00 00 65
8268:00 00 00 00 00 00 00 6D
8270:00 00 00 00 00 00 00 75
8278:FF FF FF FF FF FF FF 7D
8280:00 00 00 00 00 00 00 85
8288:00 00 00 00 00 00 00 8D
8290:00 00 00 00 00 00 00 95
8298:00 00 00 00 00 00 00 9D
82A0:32 32 32 32 32 32 32 A5
82A8:E5 E5 E5 E5 E5 E5 E5 AD
82B0:0C 0C 0C 0C 0C 0C 0C B5
82B8:A0 A0 A0 A0 A0 A0 A0 BD
82C0:00 00 00 00 00 00 00 C5
82C8:00 00 00 00 20 94 85 20 4C
82D0:A1 85 A9 79 A2 85 8D 11 2C
82D8:03 8E 12 03 38 A5 37 ED 2A
82E0:EE 82 85 0F A5 38 ED F0 7A
82E8:82 05 0F 90 08 A9 00 A2 E4
82F0:7D 85 37 86 38 20 44 E5 16
82F8:A9 0D 20 D2 FF A2 08 20 02
8300:3A 86 A9 9B A0 86 20 1E 32
8308:AB A2 06 20 3A 86 A9 CC 5C
8310:A0 86 20 1E AB A2 09 20 09
8318:3A 86 20 30 E4 4C 74 A4 CA
8320:A6 3A E8 F0 10 20 73 00 1D
8328:C9 AD D0 06 20 AC 85 4C 05
8330:3B 83 4C 93 83 20 AC 85 F3
8338:20 73 00 A9 45 A2 86 85 0E
8340:26 86 27 A0 00 84 28 20 6D
8348:B7 85 A0 00 B1 26 F0 29 D1
8350:20 73 00 D1 26 D0 03 C8 A4
8358:D0 F2 20 B7 85 E6 28 E6 03
8360:28 A5 28 CD 44 86 90 02 26
8368:B0 2C A0 00 20 CA 85 B1 CF
8370:26 D0 F9 20 CA 85 4C 47 4C
8378:83 20 73 00 A6 28 BD 87 90
8380:86 8D 8C 83 BD 88 86 8D A2
8388:8D 83 78 20 FF FF A9 00 9B
8390:F0 01 58 20 D1 85 4C FF 9A
8398:FF 20 1F 86 20 0E E2 C9 BC
83A0:AC D0 1F 20 73 00 C9 23 6A
83A8:F0 12 BD 20 82 8D EE 83 12
83B0:AD 30 82 3D 0C 82 8D E9 2E
83B8:83 4C D2 83 20 73 00 4C 42
83C0:FE 83 20 E7 85 8C EE 83 6A
83C8:8D E9 83 20 79 00 C9 23 06
83D0:F0 20 20 73 00 20 ED 85 75
83D8:90 09 AE 43 86 BD 28 82 72
83E0:A8 B0 05 20 F4 B7 8A A8 4F
83E8:A2 FF 20 0F 86 A9 FF 4C 5D
83F0:12 7D 20 73 00 AD EE 83 B3
83F8:AE 43 86 20 2A 7D 20 DA 5D
8400:85 20 ED 85 B0 0A 20 F4 CC
8408:B7 8A AE 43 86 4C 24 7D C4
8410:60 20 1F 86 20 06 86 B0 74
8418:06 20 0C 86 20 0F 7D 20 6E
8420:DA 85 20 06 86 B0 06 20 7F
8428:18 86 20 15 7D 20 DA 85 DB
8430:20 06 86 B0 06 20 0C 86 F5
8438:20 1B 7D 20 DA 85 20 06 FC
8440:86 B0 06 20 0C 86 20 1E 54
8448:7D 20 DA 85 20 06 86 B0 A2
8450:06 20 0C 86 20 21 7D 20 EE
8458:DA 85 20 06 86 B0 13 20 D1
8460:0C 86 AD 35 82 90 05 1D 97
8468:0C 82 B0 03 3D 14 82 8D 2B
8470:35 82 60 AD 97 83 8D 08 89
8478:03 AD 98 83 8D 09 03 20 70
8480:AC 84 A9 B5 A0 86 4C 1E 67
8488:AB 20 79 00 F0 1E C9 3A 6C
8490:F0 1A C9 AC F0 1E 20 F4 D1
8498:B7 8A F0 02 A9 01 CD C1 0D
84A0:85 F0 06 8D C1 85 20 09 AF
84A8:7D 4C C0 85 A9 00 8D 15 71
84B0:D0 4C 09 7D 20 73 00 4C 49
84B8:C0 85 20 ED 85 B0 06 20 81
84C0:F4 B7 8E 36 82 20 DA 85 37
84C8:20 06 86 B0 06 20 F4 B7 91
84D0:8E 37 82 60 20 1F 86 20 EF
84D8:06 86 B0 06 20 18 86 9D 09
84E0:78 82 20 DA 85 20 06 86 B7
84E8:B0 06 20 18 86 9D 98 82 AF
84F0:20 DA 85 20 06 86 B0 09 28
84F8:20 18 86 9D 88 82 9D 40 8C
8500:82 20 DA 85 20 06 86 B0 DE
8508:06 20 18 86 9D 90 82 60 1E
8510:20 1F 86 20 06 86 B0 06 77
8518:20 18 86 9D C0 82 20 DA 0F
8520:85 20 06 86 B0 06 20 18 15
8528:86 9D A0 82 20 DA 85 20 B1
8530:06 86 B0 06 20 18 86 9D 62
8538:A8 82 20 DA 85 20 06 86 29
8540:B0 06 20 29 86 9D B0 82 4A
8548:20 DA 85 20 06 86 B0 06 7E
8550:20 29 86 9D 88 82 60 20 11
8558:1F 86 20 F1 B7 8A AE 43 40
8560:86 C9 00 F0 07 C9 01 F0 82
8568:06 4C 3F 7D 4C 39 7D 4C D7
8570:3C 7D 20 F4 B7 8E 8F 83 E6
8578:60 20 AA B1 85 0F 98 05 CA
8580:0F F0 0B AC 1F D0 A9 00 0B
8588:20 91 B3 4C 73 7D AC 1E 4C
8590:D0 4C 86 85 AD 08 03 8D 61
8598:97 83 AD 09 03 8D 98 83 99
85A0:60 A9 20 A2 83 8D 08 03 D9
85A8:8E 09 03 60 A5 7A 8D B8 8E
85B0:85 A5 7B 8D BC 85 60 A9 96
85B8:FF 85 7A A9 FF 85 7B 60 7C
85C0:A9 00 D0 03 4C 03 7D 4C A0
85C8:06 7D E6 26 D0 02 E6 27 F8
85D0:60 A5 7A D0 02 C6 7B C6 BA
85D8:7A 60 20 79 00 F0 05 C9 6C
85E0:3A F0 01 60 68 68 60 20 31
85E8:8A AD 4C F7 B7 A0 00 B1 9F
85F0:7A C9 AC F0 02 18 60 20 A1
85F8:73 00 F0 08 C9 3A F0 04 79
8600:C9 2C D0 F3 38 60 20 0E E7
8608:E2 4C ED 85 20 F4 B7 18 0C
8610:8A F0 01 38 AE 43 86 60 32
8618:20 F4 B7 8A 4C 14 86 20 F1
8620:F4 B7 8A 29 07 8D 43 86 F4
8628:60 20 E7 85 84 26 85 27 B1
8630:46 27 66 26 A5 26 AE 43 BF
8638:86 60 A9 20 D2 FF CA EE
8640:D0 FA 60 00 14 4D 4F 56 4B
8648:53 50 52 00 53 50 52 49 27
8650:54 45 00 53 50 52 43 4F AF
8658:4C B0 00 41 4E 49 4D 41 3F
8660:54 45 00 42 4F 55 4E 44 BD
8668:41 52 59 00 42 4F 55 4E 1E
8670:43 45 00 53 50 49 49 54 40
8678:00 5A 41 50 00 49 52 51 64
8680:00 4B 49 4C 4C 00 00 99 4A
8688:83 11 84 BA 84 D4 84 10 68

```



```

8690:85 57 85 89 84 0C 7D 72 41
8698:85 73 84 4D 4F 42 20 4D BB
86A0:41 53 54 45 52 20 20 56 AB
86A8:32 2E 31 30 2F 39 32 30 76
86B0:33 30 39 0D 00 0D 4D 4F 79
86B8:42 20 4D 41 53 54 45 52 75
86C0:20 44 49 53 41 42 4C 45 3E
86C8:44 2E 0D 00 43 4F 50 59 76
86D0:52 49 47 48 54 20 31 39 85
86D8:39 32 20 20 42 59 20 48 15
86E0:4F 4E 47 20 50 48 41 4D 87
86E8:0D 00 00 00 00 00 00 00 7C

```

Hong Pham, the author of Pixel Mover (May 1992), lives in Antigonish, Nova Scotia, Canada.

136 COLORS

By David Kwong

As most people know, the 64 is capable of producing 16 different colors. How would you like to increase that number to 136 colors?

You can with 136 Colors. This interesting program does it by placing differently colored pixels side by side to produce a third color. Since the 64 has 16 built-in colors, it would appear that you could create 256 colors by combining the 16 x 16 color matrix. In reality, you get a total of 136 different hues, since 120 of them would be duplicated.

There are three programs built into the main 136 Colors program. The first program is an editor that will produce sprites capable of 136 colors. Additionally, each sprite character can have up to four colors simultaneously in high resolution mode. The second program is an interrupt program designed to make programming in BASIC with 136 Colors a lot easier. The third program is also an interrupt program designed to be used with other programs to make 136 Colors available for use.

Typing It In

Since 136 Colors is written entirely in machine language, enter it with MLX, our machine language entry program. See "Typing Aids" elsewhere in this section. When MLX prompts, respond with the following values.

Starting address: C79C

Ending address: CFAB

Be sure to save a copy of the program before exiting MLX.

Program 1

Load the program with the .8,1 extension, and then type *NEW*. To activate the first program, type *SYS 51200*.

The first thing to do is to select a block number, indicated at the upper right corner. A block number is an address where sprites can be stored. Recommended block numbers are 128-255 (block numbers range from 0 to 255). To find the actual address where the sprite is stored, multiply the block number by 64.

After you've selected a block number, a cursor appears in a grid that is used to create a sprite. The sprite that the grid represents is located at the upper right of the screen. The keys used to move the cursor are displayed at the lower right of the screen. Press f1 to begin drawing. A menu at the bottom provides other options. One option, NO DR/ER, means that the cursor will neither draw nor erase. This option lets you move the cursor without affecting what's on the screen.

To change colors while in draw mode, press either 1, 2, or 3. To change a sprite into its 136-color shape, either exit or change the block number. The program will then ask you whether or not to change the sprite into 136-color mode. If you elect to do so, the program then will ask you where to store the 136-color sprite.

Each 136-color sprite is composed of two normal sprites, one on top of the other. Sprite 1 is represented by color 1; sprite 2 is represented by color 2. Color 3 is divided between the two sprites. When the two sprites are overlapped, color 3 is capable of producing a color from the 136-color palette. The two sprites must have the same coordinates for them to overlap perfectly.

Program 2

The second program, which is an interrupt program, is activated or deactivated by *SYS 52600*. When activated, you'll see a message onscreen that says *136 BAS ON*.

This program provides 16 new sprite registers that will ease the usage of the four high-resolution sprites and 136 colors. There are only four high-resolution sprites, instead of the normal eight, because of the fact that each hi-res sprite requires two normal sprites.

This program defines hi-res sprite 1 as the overlap of sprites 0 and 1. Hi-res sprite 2 is the overlap of sprites 2 and 3, hi-res sprite 3 is the overlap of sprites 4 and 5, and so on.

The first eight registers from 52882 to 52889 provide the x- and y-coordinates of the four high-resolution sprites. The first high-resolution sprite can be moved by using the horizontal register 52882 and the vertical register 52883, much like the system used by the 64 to move the eight normal sprites. Therefore, every two registers provide the horizontal and vertical registers of one hi-res sprite.

The next four registers, 52890 to 52893, provide the colors of each of the four hi-res sprites. The color numbers range from 1 to 136.

The last four registers, 52894 to 52897, provide the block numbers for the four hi-res sprites.

This interrupt program supposes the block numbers for each hi-res sprite to be next to each other. Keep in mind that one hi-res sprite is composed of two normal sprites. Therefore, when you choose block number 200, the two overlapping sprites will be composed of blocks 200 and 201.

All registers are write-only registers. When you attempt to read them, they will return a 0. When the registers are 0, the interrupt program will not alter any sprites. Therefore, should you poke 52882,0, nothing will happen, meaning that if you originally poked 140, poking a 0 will not move it from location 140 to location 0.

In order to see the sprites you have produced, you must first set register 53269 to turn on the sprites you desire. Hi-res sprite 1 can be turned on with a *POKE 53269, 3*. *POKE 53269, 12* turns on hi-res sprite 2. *POKE 53269, 40* turns on sprite 3, and *POKE 53269, 192* turns on sprite 4. To turn on more than one sprite, simply add up the previous values.

Program 3

The third program is activated or deactivated by *SYS 52900*. When activated, you'll see *136C ON* printed on the screen. This simple program is designed to work with other programs that can make use of the 136 colors.

The only register provided is at

52844. This register is a 136-color register. By poking colors 1 to 136 into this register, 2 colors will be returned at locations 52898 and 52899. When the 2 colors are placed together, they'll combine to create 1 of the 136 available colors.

Since machine language programs may be too fast for the interrupt to be effective, you must keep track of location 52844. After execution of the interrupt, 0 will be stored in location 52844. If using machine language, you may choose to poke the required color in 52844 and then JSR \$CE5A (make sure the interrupt is deactivated) to obtain the two colors in locations 52898 and 52899.

Technical Notes

This program takes up minimal space from \$C79C (51100) to \$CFAA (53162). Considering that 136 Colors is composed of three programs, applications that require only one of these three programs may isolate that particular program for usage.

The first program is located from \$C79C (51100) to \$CD77 (52599), the second program is located from \$CD78 (52600) to \$CFAA (53162), and the third program is located from \$CE5A (52826) to \$CFAA (53162).

Since different color combinations may produce the same color, there may in fact be less than 136 colors. Following is a color chart of the 136 colors. The colors are organized from brightest to darkest. (These colors were very difficult to organize. Please excuse some slight mistakes!)

White-Black	(1-13)
Gray 2	(14-16)
Extra Gray	(17-23)
Brown 1	(24-32)
Brown 2	(33-35)
Brown 3	(36-38)
Brown 4	(39-42)
Red	(43-51)
Orange	(52-57)
Yellow	(58-64)
Tan	(65-71)
Green 1	(72-77)
Green 2	(78-81)
Green 3	(82-86)
Green 4	(87-93)
Green 5	(94-95)
Cyan	(96-102)

Blue	(103-111)
Purple 1	(112-118)
Purple 2	(119-123)
Purple 3	(124-127)
Purple 4	(128-134)
Purple 5	(135-136)

Seeing Is Believing

The 136 Demo program is designed to show the various colors in action and to provide programmers with additional details on how to use 136 Colors.

The demonstration consists of a BASIC program and machine language sprite data. To avoid typing errors, use The Automatic Proofreader to enter the BASIC portion. Use MLX to enter the sprite data. When MLX prompts, respond with the following values.

Starting address: 3200
Ending address: 347F

Before leaving MLX, save the sprites with the filename Sprites. When the demonstration runs, it loads 136 Colors and Sprites and looks for those filenames.

136 COLORS

C79C:A2	00	A0	00	BD	F3	CC	85	6E
C7A4:FD	BD	08	CD	85	FE	A9	0E	0A
C7AC:91	FD	C8	C0	18	D0	F9	E8	8A
C7B4:E0	15	F0	03	4C	9E	C7	A9	5E
C7BC:00	8D	52	CD	4C	62	C8	29	7D
C7C4:07	49	07	A8	4C	DE	C9	20	2B
C7CC:E4	CB	A9	01	8D	15	D0	4C	B5
C7D4:08	CB	BD	21	04	29	0F	18	20
C7DC:65	FB	85	FB	A9	00	65	FC	A3
C7E4:85	FC	F0	03	4C	0A	CB	A5	8C
C7EC:FB	8D	F8	07	A0	06	06	FB	92
C7F4:26	FC	88	D0	F9	A9	00	8D	F8
C7FC:52	CD	60	00	A9	06	8D	21	D6
C804:D0	A9	0E	8D	20	D0	8D	86	E8
C80C:02	A9	00	8D	8A	02	A9	0A	AB
C814:8D	00	D0	A9	3C	8D	01	D0	0C
C81C:A9	93	20	D2	FF	A9	01	8D	CE
C824:10	8D	0D	27	D0	8D	15	D0	CD
C82C:8D	D6	CB	A9	04	8D	D7	CB	20
C834:A9	FE	8D	B8	CB	A9	CB	8D	C1
C83C:B9	CB	20	B5	CB	A9	01	8D	91
C844:E4	D8	A9	0F	8D	5C	D9	A9	DF
C84C:07	8D	D4	D9	A2	00	BD	E4	72
C854:CC	E8	A8	A9	03	99	98	DB	C1
C85C:4C	55	CD	20	0A	CB	A9	00	F7
C864:8D	4D	CD	8D	4E	CD	A9	29	C8
C86C:8D	90	C8	A9	04	8D	91	C8	DE
C874:A9	15	8D	CE	C8	A0	00	A2	2A
C87C:00	A9	80	8D	85	C8	B1	FB	10
C884:29	00	F0	05	A9	51	4C	8F	D3
C88C:C8	A9	2D	8D	00	00	E8	E0	1E
C894:08	F0	0E	E0	10	F0	0A	E0	6F
C89C:18	F0	06	4E	85	C8	4C	5F	63
C8A4:CD	C8	E0	18	F0	0B	EE	90	0F

C8AC:C8	D0	03	EE	91	C8	4C	7D	EB
C8B4:C8	18	AD	90	C8	69	11	8D	0B
C8BC:90	C8	AD	91	C8	69	00	8D	11
C8C4:91	C8	CE	CE	C8	F0	04	4C	76
C8CC:7B	C8	EA	AC	4D	CD	AE	4E	C3
C8D4:CD	BD	F3	CC	85	FD	BD	08	AF
C8DC:CD	85	FE	AD	52	CD	F0	0F	2C
C8E4:C9	01	F0	07	A9	0E	91	FD	D0
C8EC:4C	F3	C8	A9	01	91	FD	38	D7
C8F4:A5	FE	E9	D4	85	FE	A9	80	9F
C8FC:11	FD	91	FD	20	E4	FF	F0	2E
C904:FB	AC	4D	CD	AE	4E	CD	C9	5B
C90C:55	F0	43	C9	49	F0	40	C9	E3
C914:4F	F0	40	C9	4A	F0	42	C9	94
C91C:4B	F0	39	C9	4E	F0	39	C9	C7
C924:4D	F0	3B	C9	2C	F0	36	C9	F9
C92C:31	F0	37	C9	32	F0	38	C9	A7
C934:33	F0	39	C9	85	F0	3D	C9	95
C93C:86	F0	3E	C9	87	F0	3F	C9	FB
C944:88	F0	43	C9	93	F0	45	C9	12
C94C:42	F0	4F	4C	00	C9	88	CA	EE
C954:4C	A7	C9	CA	C8	4C	A7	C9	6E
C95C:E8	88	4C	A7	C9	C8	E8	4C	1A
C964:A7	C9	A9	01	4C	72	C9	A9	EC
C96C:0F	4C	72	C9	A9	07	8D	F0	FA
C974:C8	4C	6A	C8	A9	01	4C	84	C7
C97C:C9	A9	02	4C	84	C9	A9	00	03
C984:8D	52	CD	4C	C5	C9	20	14	9B
C98C:CA	4C	03	CA	A0	00	A9	00	FD
C994:91	FB	C8	C0	3F	D0	F9	4C	92
C99C:9C	C7	20	14	CA	4C	1C	C8	3E
C9A4:EA	EA	EA	C0	FF	D0	02	A0	B9
C9AC:17	C0	18	D0	02	A0	00	E0	7F
C9B4:FF	D0	02	A2	14	E0	15	D0	06
C9BC:02	A2	00	8C	4D	CD	8E	4E	CF
C9C4:CD	AD	52	CD	FD	36	8A	0A	51
C9CC:6D	4E	CD	AA	98	29	18	F0	99
C9D4:06	C9	08	F0	01	E8	E8	98	04
C9DC:4C	C3	C7	A9	01	88	30	04	A9
C9E4:0A	4C	E1	C9	48	8A	A8	68	8F
C9EC:AE	52	CD	E0	02	F0	07	11	27
C9F4:FB	91	FB	4C	00	CA	49	FF	EC
C9FC:31	FB	91	FB	4C	6A	C8	A9	61
CA04:00	8D	8A	02	8D	15	D0	8D	5E
CA0C:10	D0	A9	93	20	D2	FF	60	F8
CA14:A9	00	8D	15	D0	A9	25	8D	86
CA1C:B8	CB	A9	CD	80	B9	CB	A9	A7
CA24:6A	8D	D6	CB	A9	04	8D	D7	3A
CA2C:CB	20	B5	CB	20	E4	FF	F0	A8
CA34:FB	C9	59	F0	07	C9	4E	D0	41
CA3C:F3	4C	05	CB	20	E4	CB	A9	12
CA44:3E	8D	B8	CB	A9	CD	8D	B9	89
CA4C:CB	A9	6A	8D	D6	CB	20	B5	34
CA54:CB	20	0A	CB	A5	FB	85	FD	FB
CA5C:A5	FC	85	FE	A9	32	8D	72	48
CA64:04	20	0A	CB	A0	3F	A9	00	57
CA6C:88	91	FB	91	FD	D0	F9	8D	F7
CA74:A3	CA	A9	40	8D	C4	CA	A9	86
CA7C:29	8D	86	CA	A9	D8	8D	87	DA
CA84:CA	AD	00	00	29	0F	AA	A9	6F
CA8C:80	E0	0F	D0	04	11	FB	91	77
CA94:FB	E0	01	D0	04	11	FD	91	7F
CA9C:FD	E0	07	D0	1F	AA	A9	00	4E
CAA4:29	01	D0	07	8A	2C	A1	CA	AC
CAAC:4C	B3	CA	8A	2C	09	CB	D0	45
CAB4:07	11	FB	91	FB	4C	C0	CA	08
CABC:11	FD	91	FD	AD	86	CA	C9	53
CAC4:00	D0	1C	18	69	11	8D	86	C4
CACC:CA	AD	87	CA	69	00	8D	87	BE
CAD4:CA	AD	C4	CA	69	28	8D	C4	4C

CD0C:D8	D8	D9	D9	D9	D9	D9	D9	E6
CD14:DA	DA	DA	DA	DA	DA	DA	DB	B0
CD1C:DB	00	00	00	00	00	00	00	A5
CD24:00	03	0F	0E	16	05	12	14	40
CD2C:20	13	10	12	09	14	05	9E	01
CD34:28	19	2F	0E	29	3F	80	BF	F7
CD3C:00	00	13	10	12	09	14	05	1D
CD44:20	23	31	80	00	10	FF	BF	E6
CD4C:00	00	FF	FF	00	00	FF	FF	E7
CD54:00	E0	0C	F0	03	4C	52	C8	6F
CD5C:4C	5F	C8	EE	90	C8	D0	03	4A
CD64:EE	91	C8	4C	82	C8	FF	FF	F7
CD6C:00	00	FF	FF	00	00	FF	FF	0D
CD74:00	00	FF	FF	AD	14	03	C9	90
CD7C:E3	D0	31	AD	15	03	C9	CD	55
CD84:D0	2A	78	AD	F0	CD	8D	14	EA
CD8C:03	AD	F1	CD	8D	15	03	58	4F
CD94:A2	00	BD	A2	CD	20	D2	FF	F7
CD9C:E8	E0	E0	D0	F5	60	0D	31	30
CDA4:33	36	43	20	42	41	53	20	AF
CDAC:4F	46	46	0D	AD	14	03	8D	6C
CDB4:F0	CD	AD	15	03	8D	F1	CD	43
CDBC:78	A9	E3	8D	14	03	A9	CD	22
CDCA:8D	15	03	58	A2	00	BD	D6	B9
CDCC:CD	20	D2	FF	E8	E0	0D	D0	67
CDD4:F5	60	0D	31	33	36	43	20	51
CDDC:42	41	53	20	4F	4E	0D	A2	6E
CDE4:00	BD	92	CE	D0	08	E8	E0	88
CDEC:10	D0	F6	4C	00	00	BD	92	76
CDEF:CE	A8	A9	00	9D	92	CE	E0	0D
CDFC:00	90	1B	E0	0C	90	35	8A	C9
CE04:38	E9	0C	0A	AA	AD	11	D0	59
CE0C:10	FB	98	9D	FA	07	C8	98	AB
CE14:9D	F9	07	4C	EF	CD	8A	29	99
CE1C:01	F0	07	8A	0A	AA	CA	4C	DC
CE24:29	CE	8A	0A	AA	AD	11	D0	FA
CE2C:10	FB	98	9D	00	D0	9D	02	3E
CE34:D0	4C	EF	CD	8A	38	E9	08	39
CE3C:0A	48	8C	6C	CE	20	5A	CE	F3
CE44:68	AA	AD	11	D0	10	FB	AD	F3
CE4C:A2	CE	9D	27	D0	AD	A3	CE	68
CE54:9D	28	D0	4C	EF	CD	A9	20	D3
CE5C:A2	69	CE	A9	CF	8D	6A	CE	E7
CE64:A2	00	A0	00	AD	00	00	C9	9E
CE6C:00	F0	17	E8	E0	10	D0	07	A7
CE74:C8	98	AA	C0	10	F0	11	EE	53
CE7C:69	CE	D0	03	EE	6A	CE	4C	D7
CE84:68	CE	8E	A2	CE	8C	A3	CE	C3
CE8C:A9	00	8D	6C	CE	60	00	00	6F
CE94:FF	FF	00	00	FF	FF	00	00	32
CE9C:FF	FF	00	00	FF	FF	00	00	3A
CEA4:AD	14	03	C9	07	D0	2D	AD	9E
CEAC:15	03	C9	CF	D0	26	78	AD	89
CEB4:10	CF	8D	14	03	AD	11	CF	02
CEBC:8D	15	03	58	A2	00	BD	CE	AB
CEC4:CE	20	D2	FF	E8	E0	0A	D0	DB
CECC:F5	60	0D	31	33	36	43	20	4B
CED4:4F	46	46	0D	AD	14	03	8D	96
CEDC:10	CF	AD	15	03	8D	11	CF	BD
CEE4:78	A9	07	8D	14	03	A9	CF	BD
CEEC:8D	15	03	58	A2	00	BD	FE	0C
CEF4:CE	20	D2						

NOVEMBER 1992 COMPUTE G-31

PROGRAMS

```

HF 290 GOSUB2000
GE 300 IFLO=CL(C,1)THEN=-1:GO
      TO270
AB 310 IFLO<CL(C,0)THEN250
AJ 320 GOTO270
KG 330 DATA "COLORS*"
FK 340 DATA "E12"
QF 350 DATA "BY DAVID KWONG*"
SJ 360 DATA "E12345678"
FX 370 DATA "{BLU}*"
DC 380 DATA "PRESS ANY KEY TO
      {SPACE}CONTINUE<"
RH 400 POKE53269,0
MH 410 PRINTCHR$(147)
HX 420 POKE52882,0:POKE52883,7
      5:POKE52894,206:POKE528
      90,129
PM 430 POKE53269,3
EE 440 FORX=0TO174STEP2:POKE52
      882,X:NEXTX
MQ 450 EN=0:SN$="":PRINT"
      {HOME}{7 DOWN}{GRN}"
JG 460 C=INT(RND(1)*136)+1
CP 470 POKE52890,C
XD 480 FORW=1TO30
MB 490 GOSUB2000
CQ 500 IFPEEK(198)>0ANDEN=1THE
      NPOKE198,0:GOTO700
GQ 510 NEXTW
AF 520 GOTO460
CS 530 DATA "IN ADDITION TO BE
      ING ABLE TO PRODUCE*"
SR 540 DATA "136 COLORS, THIS
      {SPACE}PROGRAM CAN ALSO
      *"
RK 550 DATA "CREATE 4 HIGH RES
      OLUTION (1 PIXEL RES-*)"
FS 560 DATA "OLUTION) SPRITES,
      EACH WITH 4 COLORS.*"
GS 570 DATA "OF THOSE 4 COLORS
      , 1 COLOR IS CAPABLE*"
CA 580 DATA "OF 136 COLORS. TH
      E OTHER 3 COLORS ARE*"
PA 590 DATA "RESTRICTED TO THE
      16 COLORS OF THE*"
EX 600 DATA "COMMODORE 64. EAC
      H HIGH RESOLUTION*"
GG 610 DATA "SPRITE IS CREATED
      FROM TWO SPRITES.*"
FP 620 DATA "INCLUDED WITH THE
      PROGRAM IS AN EDITOR*"
CA 630 DATA "TO PRODUCE THESE
      {SPACE}4 HIGH RESOLUTIO
      N*"
XR 640 DATA "SPRITES. THERE AR
      E ALSO TWO INTERRUPT*"
EK 650 DATA "ROUTINES INCLUDED
      TO EASE THE USAGE*"
EP 660 DATA "OF 136 COLORS AND
      HI-RES SPRITES.*"
JC 670 DATA "{BLU}*"
DB 680 DATA "PRESS ANY KEY TO
      {SPACE}CONTINUE<"
ED 700 POKE53269,0:PRINTCHR$(1
      47)
AH 710 POKE52882,138:POKE52884
      ,162:POKE52886,186:POKE
      52888,210

```

```

RE 720 POKE52883,75:POKE52885,
      75:POKE52887,75:POKE528
      89,75
MQ 730 POKE52894,208:POKE52895
      ,208:POKE52896,208:POKE
      52897,208
RP 740 POKE52890,1:POKE52891,2
      :POKE52892,3:POKE52893,
      4
FE 745 IFPEEK(52897)<>0THEN745
HH 750 POKE53269,255
MB 754 PRINT"{HOME}{10 DOWN}
      {WHT}";:EN=0:SN$=" "
ED 755 GOSUB2000
QM 756 IFEN=0THENGOTO755
EQ 760 PRINT"{HOME}{6 DOWN}
      {WHT}";TAB(15);"]"
PX 770 PRINT"{DOWN}";TAB(12);"
      COLOR"
BM 780 DIMC(3):C(0)=1:C(1)=2:C
      (2)=3:C(3)=4:D=0
PR 790 FORS=0TO3
FC 800 POKE52890+S,C(S)
ER 810 NEXTS
RJ 815 PRINT"{HOME}{8 DOWN}";T
      AB(17);"{4 SPACES}"
HM 816 PRINT"{UP}";TAB(17);C(0
      )
XP 820 GETAS:IFAS$=""THEN820
BQ 830 IFAS$="J"THEN=-1
FJ 840 IFAS$="K"THEN=1
JP 850 FORLR=0TO3
JQ 860 C(LR)=C(LR)+D
BG 870 IFC(LR)>136THENC(LR)=1
XF 880 IFC(LR)<1THENC(LR)=136
EE 890 NEXTLR:D=0
KB 900 IFAS$="E"THEN1020
PA 910 GOTO790
AR 920 DATA "NOW, YOU MAY OBSE
      RVE THE 136 COLORS*"
QP 930 DATA "YOURSELF BY SCROL
      LING TO THE LEFT BY*"
CM 940 DATA "PRESSING 'J' AND
      {SPACE}SCROLLING TO THE
      RIGHT*"
GX 950 DATA "BY PRESSING 'K'.
      {SPACE}TO END, PRESS 'E
      '.*"
XR 960 DATA "YOU WILL NOTICE T
      HAT THE COLORS ARE*"
BK 970 DATA "ORGANIZED INTO SE
      VERAL GROUPS. I HAVE*"
EP 980 DATA "ARRANGED EACH GRO
      UP FROM BRIGHTEST TO*"
AX 990 DATA "DARKEST. EACH SPR
      ITE HAS ITS OWN COLOR*"
RK 1000 DATA "ADDRESS IN WHICH
      TO POKE ITS COLOR*"
GD 1010 DATA "NUMBER.<"
FF 1020 PRINTCHR$(147):POKE532
      69,0
BR 1030 POKE53281,6:POKE53280,
      14:POKE646,14
RA 1040 END
EA 2000 IFEN=1THEN2075
ER 2010 IFSN$<>""THEN2045
GH 2020 READSN$
GK 2030 IFLEFT$(SN$,1)=""E"THE

```

```

N2000
HA 2040 L=LEN(SN$):CH=0:PRINTT
      AB((41-L)/2);
FA 2045 CH=CH+1
HP 2050 IFMID$(SN$,CH,1)=""TH
      ENSN$="":PRINT:GOTO207
      5
RM 2060 IFMID$(SN$,CH,1)=""<"TH
      ENEN=1:GOTO2075
RG 2070 PRINTMID$(SN$,CH,1);
JF 2075 RETURN
QJ 2080 R=LEN(SN$)-1
GR 2090 FORRT=1TOR:PRINT:NEXTR
      T
EB 2100 SN$="":GOTO2075

```

SPRITES

```

3200:00 2A 00 00 54 00 00 AA 3C
3208:00 01 54 00 02 AA 05 F6
3210:14 00 00 2A 00 00 14 00 49
3218:00 2A 00 00 14 00 00 2A D1
3220:00 00 14 00 00 2A 00 00 AF
3228:14 00 00 2A 00 00 14 00 61
3230:00 2A 00 00 14 00 02 AA 6E
3238:A0 05 55 50 0A AA A8 00 2A
3240:00 54 00 00 AA 00 01 54 65
3248:00 02 AA 00 05 14 00 0A 05
3250:2A 00 00 14 00 00 2A 00 5F
3258:00 14 00 00 2A 00 00 14 27
3260:00 00 2A 00 00 14 00 00 5A
3268:2A 00 00 14 00 00 2A 00 77
3270:00 14 00 00 2A 00 05 55 8A
3278:50 0A AA A8 05 55 50 00 85
3280:00 0A 00 00 55 40 02 AA C1
3288:A0 05 51 50 0A 00 20 04 52
3290:00 14 00 00 08 00 00 14 4E
3298:00 00 20 00 15 50 00 2A 16
32A0:A0 00 15 50 00 00 28 00 4D
32A8:00 14 00 00 08 04 00 14 76
32B0:0A 00 20 05 51 50 02 AA E9
32B8:A0 00 55 40 00 0A 00 00 44
32C0:00 15 00 00 AA 80 01 55 19
32C8:50 02 A0 A8 05 00 10 0A C6
32D0:00 08 00 00 14 00 00 08 DF
32D8:00 00 14 00 2A A8 00 55 09
32E0:50 00 2A A8 00 00 14 00 65
32E8:00 08 00 00 14 0A 00 08 20
32F0:05 00 10 02 A0 A8 01 55 F8
32F8:50 00 AA 80 00 15 00 00 37
3300:00 2A 80 01 55 40 02 AA 6B
3308:A0 05 00 50 02 00 20 04 59
3310:00 00 0A 00 00 04 00 00 C7
3318:0A 2A 00 14 55 40 0A AA B9
3320:A0 15 40 50 0A 00 08 14 9D
3328:00 14 0A 00 08 04 00 14 39
3330:0A 00 28 05 00 50 02 AA E0
3338:A0 01 55 40 00 2A 00 FF 86
3340:00 55 00 00 AA A0 01 55 2B
3348:50 02 80 20 05 00 10 0A BB
3350:00 00 04 00 00 0A 00 00 5F
3358:04 15 00 08 AA A0 15 55 DD
3360:50 0A 80 A8 15 00 14 0A E6
3368:00 08 04 00 14 0A 00 08 22
3370:04 00 10 02 80 A8 01 55 F8
3378:50 00 AA A0 00 55 00 FF BB
3380:00 01 F8 00 07 F8 00 7F C1
3388:FE 01 FF FC 03 FF E0 07 5F
3390:FF 00 1F FC 00 3F F0 00 89
3398:7F 80 08 FF 00 54 7C 00 2A

```



```

33A0:AA 38 01 55 20 0A AA 00 5E
33A8:15 54 00 AA A0 01 55 50 5D
33B0:02 AA A0 15 54 40 2A AA CA
33B8:00 05 40 00 0A 00 00 FF B8
33C0:00 00 00 00 00 00 00 27
33C8:00 00 00 03 00 00 1F 00 9D
33D0:00 FE 00 03 FD 00 0F FF 35
33D8:00 7F F6 00 FF AA 03 FF AE
33E0:55 07 FE AA 1F F5 54 3F F6
33E8:EA A8 7F 55 50 7E AA A0 A6
33F0:ED 55 40 8A AA 00 15 55 29
33F8:00 0A A0 00 11 00 00 FF 7E
3400:AA AA AA 55 55 55 AA AA 13
3408:AA 55 55 55 AA AA AA 55 C5
3410:55 55 AA AA AA 55 55 55 23
3418:AA AA AA 55 55 55 AA AA 2B
3420:AA 55 55 55 AA AA AA 55 DD
3428:55 55 AA AA AA 55 55 55 3B
3430:AA AA AA 55 55 55 AA AA 43
3438:AA 55 55 55 AA AA AA 00 A0
3440:55 55 55 AA AA AA 55 55 FD
3448:55 AA AA AA 55 55 55 AA 5B
3450:AA AA 55 55 55 AA AA AA 0E
3458:55 55 55 AA AA AA 55 55 16
3460:55 AA AA AA 55 55 55 AA 73
3468:AA AA 55 55 55 AA AA AA 26
3470:55 55 55 AA AA AA 55 55 2E
3478:55 AA AA AA 55 55 55 00 E0

```

David Kwong, 17, says he hopes this expanded palette program will generate many new ideas and give the 64 a new look. He lives in Edmonton, Alberta, Canada.

TUNNEL TRAP

By Danny English

In the days of knights and castles, disputes could be settled by a sword fight, a joust, or a good game of Tunnel Trap. The first two activities have pretty much faded into obscurity, but you can still enjoy this game for the 64.

Challenge a friend to a heated battle inside a 32-screen maze of tunnels. Destroy your opponents by slingshot or by strategically set traps. Tunnel Trap features a realtime split screen and responsive controls.

Getting Started

Tunnel Trap is written entirely in machine language. To enter it, use MLX, our machine language entry program. See "Typing Aids" elsewhere in this section. When MLX prompts, respond with the following values.

Starting address: 0801

Ending address: 1990

Be sure to save a copy of the program

before exiting MLX.

The Challenge

When you're ready to play, connect two joysticks to the computer. Although Tunnel Trap is written in machine language, it loads and runs like a BASIC program. When the title appears, you have the option of turning trap sensors on or off. Pressing f1 will enable trap sensors, and pressing f3 will disable them. They will be explained later in the article. Pressing the space bar begins the game.

The Split Screen

Playing Tunnel Trap can be a bit confusing at first. The top screen belongs to player 1, and the bottom to player 2. Each player is controlled by joystick, and each player has a status line.

The two views represent windows on different sections of a large maze. The two players begin their search for each other at opposite ends of the maze. Players control their knights with joysticks. Pressing the fire button launches slingshots. The shot fires in the last direction that the player moved. When the players enter the same screen, an image of each player appears in each window. The best way to avoid confusion is to look only at your own window.

The Deadly Traps

Besides being able to shoot at each other, each player begins the game with 25 traps. Player 1 can dig a trap anywhere in the tunnel by pressing f1; player 2 presses f7. Your enemy cannot see the traps you set, and you cannot see his. You cannot fall into your own traps. On the title screen, you have the option to enable trap sensors. These are state-of-the-art warning devices. When they're activated, a green light at the far right of the screen flashes a warning when you're near an enemy trap. The sensor won't pinpoint the trap's exact location, but it does warn you to take caution.

How to Win

On the left side of each player's status bar is a green stamina indicator. Each time a player is hit with a slingshot or falls into a trap, he loses one stamina point. When all points are gone, the oth-

er player wins that round. The game continues until someone wins three rounds. The victorious knight will be crowned champion of the day. To return to the title screen at any time, press the Commodore key in the lower left corner of the keyboard.

TUNNEL TRAP

```

0801:0B 08 70 17 9E 32 34 30 6E
0809:37 00 00 00 20 20 20 20 96
0811:20 20 20 20 20 A0 C4 B9 06
0819:3C 08 99 F8 00 B9 FD 08 F6
0821:99 33 03 88 00 F1 A0 09 4C
0829:B9 0C 00 99 FF 03 88 D0 A1
0831:F7 A9 3C 85 2D A9 20 85 5D
0839:2E 4C 00 01 12 F0 00 3C 14
0841:20 07 18 B9 6E 09 99 E8 75
0849:07 C8 D0 F7 EE 02 01 EE 19
0851:05 01 C6 F9 D0 ED A2 03 23
0859:20 34 03 00 33 C9 07 D0 95
0861:16 A2 01 20 34 03 D0 0A A0
0869:A2 04 20 34 03 18 69 07 65
0871:10 05 A2 0A 20 34 03 85 1D
0879:A8 A5 A7 85 A9 A5 FE 85 FB
0881:F7 A5 FF 85 F8 20 6C 03 73
0889:A5 F8 85 FF A5 F7 85 FE 72
0891:E8 20 34 03 00 1E A2 00 21
0899:20 34 03 A0 00 84 A0 85 2A
08A1:A6 18 A5 FC 65 A6 85 F7 58
08A9:A5 FD 65 A7 85 F8 20 6C EF
08B1:03 4C 13 01 E8 20 34 03 FB
08B9:D0 1C A0 03 84 A8 E8 20 36
08C1:34 03 F0 00 A2 08 20 34 F4
08C9:03 4C 5C 01 A2 0C 20 34 C3
08D1:03 E6 A7 4C 5C 01 E8 20 AF
08D9:34 03 D0 0A E8 20 34 03 B2
08E1:18 69 04 A8 D0 D6 E8 20 37
08E9:34 03 D0 0A A2 02 20 34 21
08F1:03 18 69 06 D0 ED A2 08 A2
08F9:20 34 03 D0 E6 A9 00 85 F7
0901:A7 A4 FB F0 CA 06 FA 2A 37
0909:26 A7 C6 FB 0C D0 F2 A8 D8
0911:60 48 B1 FE 85 FA A9 08 FE
0919:85 FB 68 A4 FE D0 02 C6 4A
0921:FF C6 FE C0 F7 D0 DE A4 B5
0929:FF C0 07 D0 D8 A9 37 85 BA
0931:01 58 4C 10 00 A4 A8 F0 79
0939:22 A5 F7 38 E5 A8 B0 03 7E
0941:C6 F8 38 85 F7 A5 FC E5 8A
0949:A8 00 02 C6 FD 85 FC B1 3A
0951:F7 88 91 FC 98 D0 F8 C4 42
0959:A9 F0 0A B1 F7 C6 FD C6 76
0961:F8 C6 A9 10 EC 60 78 E6 98
0969:01 4C 16 00 60 00 00 08 73
0971:0A 00 9E 32 30 36 34 E3 26
0979:02 45 00 78 A9 34 07 3E CF
0981:A2 05 BD 42 00 9D 2D 03 16
0989:8F 10 F7 9A B7 75 C6 32 63
0991:CE 2C 00 B1 31 E0 0E 00 5B
0999:E2 F8 A5 32 C9 08 A0 67 4A
09A1:B9 48 08 B6 01 2E EB C5 6D
09A9:66 01 08 FD E8 3C 20 ED 76
09B1:00 01 2A 2A 29 07 AA BD 64
09B9:1A 01 BD 18 C6 97 29 1F 52
09C1:AA 3A 8B 4C FF 01 A4 43 7E
09C9:AB 79 58 3B 3F 29 92 93 26
09D1:C4 60 13 13 E2 F0 C5 A9 02

```


PROGRAMS

```

09D9:87 EF A9 46 74 EB 82 73 11
09E1:E2 F0 A8 05 B2 20 83 01 25
09E9:1D C5 C8 F5 3C 23 F1 30 F6
09F1:8F 86 39 2D AA 4C 22 01 33
09F9:20 71 01 02 70 99 E6 2F 9F
0A01:AC 82 30 E6 2D D0 02 E6 18
0A09:2E 82 EB ED C6 39 10 E9 56
0A11:E8 50 2C DA 01 A9 37 85 83
0A19:01 58 20 00 40 4C AE A7 FF
0A21:CF 0D 84 EE A9 04 2C A9 3F
0A29:08 85 FF B1 2F 91 2D C8 A0
0A31:C4 FF E8 2A 10 2D 21 11 EF
0A39:2D A5 2E 5D 44 2E A0 85 8A
0A41:08 E5 4C 2F 65 FF 85 2F B4
0A49:A5 30 0B 70 F9 30 4C 00 CD
0A51:01 B9 00 EF 99 00 FF C8 E8
0A59:D0 F7 CE DC 01 CE A3 BA C0
0A61:AD DF 01 C9 DF 80 66 2C FA
0A69:9F FE 00 90 13 7B 5C C7 1D
0A71:48 1C 02 05 71 BD 33 00 2A
0A79:18 1E 4A 1B 26 1F 1E 1A 20
0A81:2C 0D 42 E1 B0 B1 38 D0 E2
0A89:28 D5 30 33 D2 E1 3A 43 36
0A91:B1 78 23 CE 20 08 44 62 F9
0A99:AF D2 98 9C 78 68 34 34 19
0AA1:08 85 C3 C6 42 1C 13 19 C1
0AA9:87 0C 80 14 18 EC 38 C5 80
0AB1:21 5C 20 45 1E 3B 02 73 1B
0AB9:C2 22 21 6D B8 21 07 A2 AD
0AC1:2A 6D 41 0F 23 0A 20 67 48
0AC9:07 80 30 96 B1 21 1A D2 0A
0AD1:08 C0 44 61 51 EC C4 BC 3D
0AD9:80 2D 17 52 C7 44 86 CA 8A
0AE1:86 33 8D 34 6C 70 99 34 87
0AE9:0A 7E 80 24 00 08 21 3D 94
0AF1:52 09 00 C8 8C B2 4F A5 71
0AF9:24 3C 80 00 19 07 0B 3E 78
0B01:88 40 21 7C 20 02 BC 22 FB
0B09:3C 20 02 66 A2 21 E7 AE 04
0B11:21 18 85 40 00 26 C8 EF 8C
0B19:96 64 49 A3 6D 36 04 08 4B
0B21:8D E6 03 A2 00 CA D0 FD 0D
0B29:CE F0 CF E1 70 A0 B9 ED 5D
0B31:67 27 AB C4 99 90 D9 88 D1
0B39:39 C8 D0 F0 58 2C 08 99 64
0B41:91 D9 99 31 DA 13 3C D0 49
0B49:30 89 99 3F A9 57 0F 8D 15
0B51:1D BC 80 C6 17 39 00 4A 89
0B59:A9 1F 8D 27 7C 29 90 87 61
0B61:8C 96 D2 40 6E 28 E6 2A CD
0B69:41 1E E0 60 30 46 8A 18 91
0B71:07 2B E6 2D 40 00 19 2C 29
0B79:D0 8D 2E D0 A9 E5 8D FC 2B
0B81:8B 0B FD 88 FE 07 8D FF 97
0B89:00 6E 00 B9 43 03 AD BA 13
0B91:60 10 02 AD BB 56 00 20 4E
0B99:05 BC F2 01 20 0F BD 42 CA
0BA1:05 AD BE 40 85 04 AD BF D8
0BA9:20 BC 02 D0 AD C0 40 40 7D
0BB1:A3 86 60 4A 5B 6C 3A 07 8F
0BB9:7B 6C DA C9 20 E4 E4 52 51
0BC1:F0 4C 39 53 F0 03 DB 06 11
0BC9:94 27 B1 14 72 90 64 38 42
0BD1:6C 90 80 C3 46 08 61 90 34
0BD9:C4 05 43 B2 01 3A 1A 24 70
0BE1:19 0E 1B CF 84 41 12 0B C0
0BE9:0F 49 0A 3C 24 1B 7E A1 0B
0BF1:41 92 E1 B0 41 02 0E 1B DD
0BF9:24 C0 0D 49 50 3C 15 19 12
0C01:D3 03 0F 4F 0D 10 0E 18 77
0C09:69 5B 0E 1B 38 0D CF AD 63
0C11:C1 15 C0 A0 BE BC D0 0B 07
0C19:02 2C 64 E4 85 C6 C6 02 EE
0C21:A6 02 02 E6 05 03 3C 0A 72
0C29:19 EB 90 17 76 24 43 5E 75
0C31:C8 21 51 E4 05 1C 12 E4 10
0C39:39 12 1B D7 49 E4 05 1C 57
0C41:05 45 5E 10 3D C5 41 46 C3
0C49:5E 68 D4 00 91 17 D3 40 16
0C51:5A 6E 80 91 17 48 F8 3E 65
0C59:79 41 2C 14 05 C5 42 2D 36
0C61:53 11 20 05 20 21 59 A1 95
0C69:3F 42 C6 03 98 17 02 E6 C6
0C71:03 5C 5E 58 78 38 40 5E F6
0C79:D0 81 01 11 1B 0A 90 17 C4
0C81:76 60 C1 F4 61 F4 14 79 F4
0C89:61 E1 61 73 D1 58 64 B4 9B
0C91:6A 58 14 41 4C C3 64 01 C3
0C99:7C 87 20 67 A5 2E 18 2A 8C
0CA1:19 82 20 62 4C BB 41 13 F7
0CA9:01 32 91 8A 33 91 B1 14 01
0CB1:18 13 A8 41 55 5E 10 B1 B9
0CB9:40 05 CB 70 2D 08 8D 58 B0
0CC1:11 42 95 41 20 B3 2A D6 37
0CC9:42 20 F9 42 4C 17 43 20 D3
0CD1:5C 24 2C A7 28 C3 00 F2 64
0CD9:42 43 A4 93 1E 91 0B 0F 0E
0CE1:3C C9 70 02 CA 30 8B BC 8E
0CE9:D0 18 29 AC 42 7F 42 CB C0
0CF1:2A EA E8 88 23 2F 34 3A 76
0CF9:8A BC D0 E8 D8 68 E9 A8 14
0D01:A3 06 F2 02 0E 9E EE 04 B9
0D09:B9 B6 02 51 42 6B 2B C0 D9
0D11:46 B0 62 32 12 00 76 F2 5A
0D19:C2 8E 4F 23 9C 8A 06 21 90
0D21:AC 01 9A 9F 8C BC C0 0A 02
0D29:FC B9 AD 27 06 F2 82 C4 9C
0D31:43 21 8E 27 C8 0B 42 00 70
0D39:EE 84 C9 78 CA 0B 38 A6 D5
0D41:83 40 02 33 0E CF 8A 23 0E 71
0D49:D0 82 EE 43 38 E2 C8 0B 68
0D51:8C 8A 22 2F 34 38 8A BC DF
0D59:D0 E0 CE 10 01 30 C8 30 79
0D61:10 AD 08 31 33 C8 0B 0C E1
0D69:9D BC 90 1D 41 E6 6F 21 0B
0D71:00 67 A1 30 05 04 04 A6 83
0D79:04 E0 FF 75 2A 28 41 21 BA
0D81:E1 47 61 8E 52 48 C8 6E 27
0D89:42 52 43 22 E3 47 A2 E3 49
0D91:91 D8 E4 91 D8 58 3D E1 E4
0D99:86 30 E1 98 82 A9 E2 AC F5
0DA1:9E 8C 83 04 F0 09 BC 7B 7F
0DA9:02 79 FC 17 4C C0 F8 40 CB
0DB1:AD 73 03 0F C9 0E F0 56 8F
0DB9:C9 0D F0 64 C9 0B F0 32 EE
0DC1:C9 07 F0 3E C9 0A F0 12 EE
0DC9:C9 06 5E F0 86 09 F0 16 75
0DD1:C0 11 25 18 A9 E0 77 76 EC
0DD9:C4 0C 2C 14 05 C5 42 A7 8B
0DE1:53 8B 20 05 20 9B 44 AC 21
0DE9:B9 44 C6 05 A0 FF 79 BC CF
0DF1:E6 05 81 16 E2 31 60 A0 8F
0DF9:01 74 28 22 3C 76 22 04 DC
0E01:44 A0 2C 22 F9 43 20 2F 5B
0E09:62 41 60 0B AB B8 2C 8C 88
0E11:9E 20 C1 40 AE BA 43 60 08
0E19:AA 2F C2 A2 ED B1 C8 68 09
0E21:C9 B0 28 82 4C 3D C8 04 68
0E29:42 2C E1 72 54 A0 00 54 4E
0E31:32 04 20 C5 4C 35 82 26 2A
0E39:01 64 22 04 44 4B 0F 34 15
0E41:26 22 44 B6 32 8D 88 05 CA
0E49:2A 43 B9 18 69 08 8D 28 B2
0E51:23 4C 0F 44 20 2D 54 50 E6
0E59:45 20 73 45 4C 91 45 40 56
0E61:B8 40 A4 4E 51 86 38 ED 68
0E69:01 89 5C 0D 45 78 89 54 38
0E71:78 F0 C9 E8 11 50 86 18 BC
0E79:6D 09 40 8D 81 8E 08 8C 52
0E81:54 05 20 F9 44 AC F0 CF 33
0E89:DC F4 21 1D 16 F3 50 67 CF
0E91:74 14 33 CE 39 1D E2 3B 7F
0E99:1D 75 D4 C9 4A 50 66 06 3F
0EA1:CF EE 05 CF A0 26 20 CB A8
0EA9:44 88 D0 FA 4C 06 47 FC D9
0EB1:3C 26 23 61 01 DC 29 10 CD
0EB9:33 41 AD 11 CF F4 9C 32 44
0EC1:32 0B AB C2 15 7E 20 F3 32
0EC9:4E BA 10 F8 B9 AD 27 06 26
0ED1:68 0D 1A 6A 14 64 36 84 72
0ED9:4E 31 0E 2D 60 CE 4E 21 F9
0EE1:A0 90 0C 60 EE 5E 01 C8 B5
0EE9:C9 F0 AC 34 0C 8E AF 22 1C
0EF1:7E 0F 18 56 E0 D8 18 45 59
0EF9:82 EE 43 38 E2 58 01 8E 08
0F01:4E B0 2D 8E 01 F0 D1 F8 69
0F09:38 CE 72 04 C0 20 FF F0 FC
0F11:10 8F A4 C9 05 90 01 60 19
0F19:AD 0D CF E1 72 B4 60 4C DC
0F21:42 41 AD 00 41 F0 02 04 3C
0F29:29 02 90 02 06 29 08 90 04
0F31:02 0C 29 0A D0 8D 0E 40 32
0F39:04 01 E9 00 90 05 06 03 7E
0F41:41 40 07 76 2E F8 64 A0 17
0F49:AD 70 96 4A 63 A0 18 C1 61
0F51:D0 69 26 04 0D 41 0B D0 8B
0F59:38 E9 78 50 B2 AD 61 54 85
0F61:09 06 A9 33 BE 82 A9 FF 41
0F69:8D 3C 57 C3 C0 46 4C 06 42
0F71:47 DD 07 A2 E8 C8 CF 46 06
0F79:AB E8 C2 C3 00 40 8A 04 56
0F81:A2 C8 F2 8A 78 72 8A 15 E1
0F89:48 C8 20 AD 05 A8 02 B0 5C
0F91:AD 06 40 B0 86 82 B0 15 CF
0F99:CA 52 40 12 A5 40 1D 29 6C
0FA1:40 51 82 15 40 4C 15 47 7A
0FA9:44 01 30 08 01 FA 84 FB A9
0FBI:A0 06 A2 58 03 6E A0 07 95
0FB9:A2 20 20 4C 47 20 DA 49 B3
0FC1:4C 39 48 C8 D0 F3 07 8E 19
0FC9:79 80 97 20 04 AC 64 03 58
0FD1:7E AC 25 AE 42 0A 20 8B EF
0FD9:47 AD B0 00 C4 8D 08 AA 34
0FE1:03 09 CF A2 EE 10 01 AD 4F
0FE9:07 CF C9 04 D0 DA E0 89 36
0FF1:19 5F 90 58 86 FE 84 FF 45
0FF9:AC 03 F7 AB 40 86 FC 84 83
1001:FD AA E0 00 F0 11 70 5C EF
1009:25 32 47 A5 CA 4C 9A 47 8F
1011:4E 00 C1 B1 FC 91 FE C8 A0
1019:C0 0A D0 F7 A5 FE E4 0C B4
1021:28 85 FE A5 FF CA FF D0 EC
1029:E0 1A 0A 0E AE D1 E8 E0 D1
1031:05 D0 D6 EB D8 20 0E 04 09
1039:8D 43 E1 01 0E A9 DA 59 63
1041:17 A9 06 74 67 58 4C 02 96
1049:50 00 53 10 0C 19 10 41 23
1051:FA 4F 9B 59 78 D4 C2 D1 3A
1059:1C 23 1B 1E 43 30 92 31 C6
1061:04 23 22 3B 46 52 B0 14 35

```


1069:A9 0B 91 F8 77 70 A1 ED 91
1071:7A 84 E7 76 02 83 4C 46 51
1079:12 DC 11 91 0F D9 1F 91 C4
1081:90 1D 91 68 48 20 EB 47 CB
1089:7B D7 0B 43 32 10 20 FA FF
1091:47 A5 F9 C9 DB D0 F1 A5 46
1099:F8 C9 E8 D0 EB 60 A8 AE B3
10A1:91 FA 4C 8B 5B 26 C0 11 91
10A9:40 B5 48 6D D5 64 40 CD C5
10B1:04 03 95 A1 5C E1 01 75 43
10B9:A8 0F 41 02 0E 53 03 4E 4C
10C1:54 1C 02 AD C2 00 51 DA C1
10C9:F1 38 42 DB 05 60 AD 14 10
10D1:22 41 F0 03 4C 06 49 82 31
10D9:DA B0 D0 64 19 C0 9A 03 F7
10E1:30 0E C5 F6 85 00 4F 62 F2
10E9:13 4F 96 2C EC 01 41 04 EE
10F1:0E 23 A5 05 50 1E A1 CE F3
10F9:23 AD 20 0C C9 AF 84 CE C0
1101:2A 56 79 B9 8D 2B 06 62 20
1109:43 E1 52 18 0D 05 71 D1 42
1111:A5 4B 40 F2 4B A5 4C 79 13
1119:54 4C 8D 42 46 8E C2 A0 E0
1121:28 20 54 49 88 C0 00 3C 02
1129:1C 77 30 FC 04 23 AC DE F1
1131:D9 0A 2B 9E 40 17 E4 88 C2
1139:DB 08 BB F0 88 DA 08 AB 3D
1141:83 61 C3 83 01 A8 36 18 5D
1149:32 06 33 C2 36 06 33 C2 8B
1151:14 76 C2 32 38 84 2E 06 CC
1159:A9 08 8D 2E DA E0 7E 83 C1
1161:CF 63 09 04 F2 3C D0 08 D7
1169:64 41 A2 52 20 74 3C A0 73
1171:01 B4 03 E4 A1 20 3C 4A 40
1179:C7 B0 E1 C1 90 FF C3 50 60
1181:47 08 30 5A 49 48 46 5E 4B
1189:F9 A2 53 47 12 CF 4D 47 DF
1191:4A 98 04 19 1D 4D 4D 03 CC
1199:5C F8 38 E9 01 85 F8 A5 83
11A1:F9 E9 11 BF F9 92 E0 22 57
11A9:24 C7 98 87 26 A8 85 12 4C
11B1:2A 3C 12 EE C8 B1 F8 C9 F1
11B9:20 D0 03 E0 21 A0 29 68 D4
11C1:4C A9 B0 04 1D 60 A5 4D CD
11C9:18 69 07 85 4D A5 4E 18 41
11D1:AF 1E 4E A6 23 A2 00 2B 56
11D9:8A 42 08 C8 24 32 C0 07 D1
11E1:D0 F7 33 A2 E0 31 E0 64 2D
11E9:D0 EA C2 29 96 4A E0 00 B9
11F1:F0 07 20 9F 4A CA CE 31
11F9:4A E6 56 A2 00 CF F5 10 2B
1201:22 03 04 8D 10 C2 31 11 4F
1209:08 1C 8D 01 41 02 2F 10 7A
1211:D7 8D 72 5E AC 04 8D 09 52
1219:96 0B 80 08 0D D0 8D 0F B1
1221:A0 78 09 BF 50 A3 40 C3 38
1229:68 08 D1 02 D3 F4 C4 5D 37
1231:D3 93 E1 B0 D1 B3 31 50 7B
1239:F4 C0 01 F4 C4 97 D3 13 B6
1241:04 4F CF 04 4E 4F 1D 6C CB
1249:A0 31 04 C9 53 DC 23 20 9B
1251:80 48 A0 E6 B0 0E 20 33 7B
1259:4F A0 E7 AD 71 0F C8 C8 47
1261:41 20 92 4B A0 E9 9E 4B 6A
1269:98 6B BA AE CD 20 CB 4A C7
1271:A0 34 62 0B 91 4D 85 0A C6
1279:44 4E A9 93 02 43 8C 82 FB
1281:8C 2E F1 60 8C F9 07 8C A2
1289:FB 07 C0 99 1E D0 C9 A5 8C
1291:F0 0A C9 5A F0 11 8C EB 60

1299:6D 0C 08 A1 90 0B 46 4C 1C
12A1:CC 49 03 26 69 20 91 43 73
12A9:4C DF 4B AC 05 3F 80 87 CD
12B1:D0 CE 11 AD 12 20 89 3B 4E
12B9:60 AC BC 03 E0 07 30 06 8A
12C1:CE 4E AD 13 80 F4 00 F0 90
12C9:1B 8D 01 1E 8D 24 01 46 26
12D1:8D 9A 04 60 AD EE 2B 80 E9
12D9:84 EF CF 8D 0E 02 C0 41 4A
12E1:AD CA 05 C0 18 33 4C 23 85
12E9:4C EE 5A 01 AD 1A 06 C9 F6
12F1:B3 F0 74 20 F2 4B E3 2D 76
12F9:01 47 2F 60 74 04 C6 4B E8
1301:85 47 A7 10 0A A1 4C FF 21
1309:4B D7 94 B1 D3 94 E1 01 2E
1311:D3 94 66 03 03 98 D0 B4 14
1319:E3 81 00 4D F8 F3 34 FA 61
1321:49 72 93 34 89 4C 20 FA F6
1329:65 DA B2 8D E9 05 AD 04 AA
1331:CF CD 05 CF F0 08 A9 02 4D
1339:39 3F 4C 84 A8 60 0A 64 DF
1341:16 88 0A D1 90 10 39 C6 F0
1349:01 D0 03 4C 6F 3F 63 18 A0
1351:65 AD 60 10 C9 E9 00 A9 52
1359:E9 8D F9 07 8D FB 07 4C 3E
1361:C0 4C A9 EA 4C D7 4C AD E6
1369:17 C7 BA 46 06 EE 8E 08 D9
1371:40 4B 0B 09 DE 39 64 35
1379:14 1E 00 AD 14 9D 41 0A BB
1381:AD 15 9E 01 F0 0B 4C 1E 12
1389:85 1E 0E 05 4C 07 4D A9 E8
1391:54 8D 2E 06 AD 64 F2 04 54
1399:C9 50 00 01 54 00 8D 16 B2
13A1:8A 20 DA 9C 04 AD 4A 20 C5
13A9:19 40 00 BA 46 20 3B 40 85
13B1:4C 8C 40 13 EC 11 39 90 09
13B9:12 B4 9F C1 CC 30 20 49 D6
13C1:7C 11 43 48 04 50 23 2C B1
13C9:21 0D 2D 85 A8 5D 2E 13 EB
13D1:EE 11 22 92 05 E3 F8 70 3B
13D9:15 21 41 F2 57 92 2E 50 DB
13E1:4C 41 59 45 52 F6 1C 54 F8
13E9:49 EA 06 56 A0 A4 54 59 2F
13F1:59 20 47 41 4D 45 E2 02 11
13F9:48 1F E4 20 E9 94 5B 05 00
1401:25 46 31 2F 46 33 5D 20 40
1409:20 9F 54 52 41 50 0E AE AF
1411:45 4E 53 4F 52 53 3A 20 43
1419:4F 41 BC 9D E3 11 21 9E EE
1421:E8 20 62 5D 50 52 45 53 91
1429:53 20 53 50 41 43 45 20 34
1431:54 4F 20 42 45 47 49 4E A7
1439:C8 C1 43 A9 A0 99 90 A0 66
1441:66 C7 63 04 5D 1B 82 3E D5
1449:4D 65 37 3C 95 40 31 0E 3A
1451:0F 45 A9 20 25 91 01 23 1E
1459:B1 91 F1 2D 07 20 F2 06 74
1461:8D F3 06 A0 28 17 CF 15 6A
1469:07 83 28 C1 42 85 07 66 B3
1471:70 DB 75 28 24 98 A9 1D ED
1479:95 2C 10 D8 8D 16 D8 46 C3
1481:20 4E 0B 84 7C 8D 86 02 20
1489:A9 93 84 32 A9 05 00 22 C3
1491:25 0D 8D 23 D0 01 22 15 57
1499:D0 54 6C 5A 82 8D 21 D0 CF
14A1:D4 49 02 34 05 28 04 B9 94
14A9:F2 62 99 18 05 C2 19 F0 EE
14B1:D0 EF 4C B9 73 4D 20 D2 47
14B9:FF 9D 87 87 D0 F5 4C A5 4F
14C1:38 88 4E 07 03 C6 40 41 57

14C9:4E A5 C5 C9 3C F0 0B C9 5D
14D1:04 F0 0C C9 05 F0 10 A7 0A
14D9:33 85 C6 60 A9 01 20 19 86
14E1:6C F0 83 20 2E 4E 4C B1 E3
14E9:78 1D 60 BC 68 00 65 18 93
14F1:19 2F 3E 32 81 7C 3E 0A E1
14F9:72 3E 4A 50 7C 3E 32 07 81
1501:7C 3E 36 60 20 E7 4E A9 AC
1509:14 BC 43 E6 21 8D 04 23 AD
1511:0C 8D 05 30 82 06 1C 02 AE
1519:0F A1 84 06 32 8D 82 D8 C9
1521:8D 00 D4 8D 01 D4 A9 03 37
1529:E1 86 CE 8D AD FF 00 59 5F
1531:01 D0 E8 9A 19 10 29 FE 31
1539:90 54 41 29 FB 44 44 D0 C5
1541:FA D6 84 A9 D0 90 A9 30 16
1549:05 7D A0 00 B1 FA 91 E8 EE
1551:FA 84 30 FA A5 FB E4 31 E7
1559:FB A5 FC 18 69 01 85 FC 63
1561:34 BC 69 40 86 FD A5 FD 7B
1569:C9 38 D0 DA B9 5C 5F 99 E5
1571:08 32 98 63 A0 D0 F5 A5 4F
1579:01 09 04 85 01 AD 18 27 55
1581:09 01 8D 0E 3C 50 10 A8 F3
1589:99 12 C8 C0 3F D0 F8 60 B9
1591:AD 8D 02 C9 02 F0 01 60 09
1599:4C E2 90 AE C1 8D 18 03 17
15A1:A9 FF C3 35 45 85 C6 20 5A
15A9:70 4F 20 86 06 70 01 5A 9A
15B1:4E 00 2C 4D EE 16 AD 10 A0
15B9:04 C9 10 90 17 A9 00 8D 50
15C1:F3 CF 20 16 E4 02 54 82 99
15C9:CE 41 20 48 44 20 A5 4B 6D
15D1:20 D7 4F A9 02 20 0B 40 6D
15D9:20 35 60 5D 67 43 20 A7 73
15E1:42 00 A3 00 F8 AF 82 08 3D
15E9:E1 00 21 45 20 F4 48 20 25
15F1:5B 46 20 E5 4C 4C FC 4F 9A
15F9:8F 1A 00 22 80 F5 0E E3 70
1601:AF 26 DB 00 EA FE FE C4 20
1609:28 00 54 94 A4 A4 E8 E8 8F
1611:F8 E3 6A 2D A0 EE 56 55 95
1619:55 FF BF BF AF AB AB AA 12
1621:56 FE FC 60 42 BC A8 47 75
1629:65 EA A5 9D 7A 64 D5 54 B6
1631:26 00 3D 02 F8 74 65 45 BC
1639:74 17 62 9D 51 65 57 55 AF
1641:4E E9 91 16 95 5D 45 15 64
1649:90 67 9C 50 E4 D4 79 5D FA
1651:16 16 07 09 55 06 19 55 C9
1659:66 6B 75 E5 51 10 10 04 8F
1661:44 48 5D E5 65 75 65 A8 40
1669:C2 EA D5 C5 21 99 CC A8 F8
1671:2D FF D7 5A 5A 6A 6A DB 02
1679:8B 21 1E 08 FE 10 B0 98 2A
1681:A4 64 BC 58 01 09 A2 25 CC
1689:1A 81 84 C2 30 4E 46 00 27
1691:9A A2 42 08 54 46 32 12 AE
1699:28 9A A1 22 04 41 8A 3D 4E
16A1:9B 35 88 89 23 0D 41 46 A8
16A9:03 03 28 3B C3 03 B2 C1 22
16B1:03 9E 0B E3 01 A2 60 27 21
16B9:EA 1E 38 29 04 40 0C 05 37
16C1:81 EE 0A 0A B8 03 2D 88 01
16C9:CD 08 09 08 0A B0 25 11 EE
16D1:49 00 41 0A 0C 5A 2D 92 22
16D9:84 62 03 0C 8A 4E E2 E3 39
16E1:0C 27 00 3C 09 69 12 A3 57
16E9:60 30 11 30 09 45 EA 8A 35
16F1:81 E2 04 E6 20 61 E3 04 D8

PROGRAMS

```

16F9:12 0E 12 E8 64 02 12 12 E4
1701:0D 33 A7 58 03 3C 05 79 89
1709:08 93 A1 22 0F 2C 12 37 FA
1711:99 28 12 0E 0F 28 A1 0C A1
1719:49 0F 0F 0D 00 C0 03 08 73
1721:81 92 02 0C 0C 0D 72 A7 D6
1729:0B 61 00 0D 24 02 0B EB 31
1731:33 11 09 E8 0A 60 04 60 27
1739:08 11 2A 09 C8 07 90 04 0D
1741:02 A2 27 0D 5C 05 0A A0 7A
1749:04 A2 2D 02 00 10 01 04 2E
1751:01 E1 09 02 0D 02 01 E3 10
1759:04 26 05 47 00 0B 04 04 60
1761:E4 0C 2A 13 13 20 3E 06 17
1769:00 01 01 80 14 E3 13 A1 F7
1771:0F 91 C1 A2 C1 0F 79 4B
1779:90 22 00 FB 0A 99 FF 23 12
1781:50 11 80 05 10 0F C2 A1 60
1789:60 E1 C0 23 4C 42 36 B7 3A
1791:73 AC D5 28 B3 36 07 4C B2
1799:87 47 41 1C 24 71 C0 54 04
17A1:22 F8 0C 04 88 44 25 CE 4F
17A9:54 06 C7 32 3C 63 77 74 72
17B1:C5 01 51 B0 ED E5 18 26 95
17B9:ED A0 6D 94 04 06 50 C1 98
17C1:B7 C1 75 80 31 20 2C 27 7C
17C9:4A 68 1C 81 D2 02 8E 41 CF
17D1:29 4E CF 06 0D 14 EA 48 65
17D9:48 1C 03 10 92 34 33 76 D6
17E1:E1 C1 B3 21 92 51 00 84 58
17E9:48 E4 8E C8 2E 4E 41 41 42
17F1:B0 71 0D C4 20 10 11 F0 17
17F9:B1 C1 08 18 19 46 82 01 DB
1801:1E 73 70 6C 64 E7 E4 AC 2B
1809:44 C9 24 E5 30 17 1B 5E 23
1811:21 32 08 2B 0C C1 86 38 BE
1819:43 20 50 C1 40 EC 01 80 51
1821:2B 20 21 63 07 76 FF 00 5B
1829:32 FE 00 46 C2 06 40 14 59
1831:13 4B 39 1D 4B E3 47 3E 6D
1839:99 29 A0 00 04 30 91 61 F9
1841:00 01 10 48 20 7E 47 32 F3
1849:02 03 45 47 80 E0 43 52 B8
1851:C6 10 DC C6 81 05 0D 17 42
1859:1E 06 49 42 00 87 1B 65 21
1861:1B 32 20 71 E1 81 B1 31 70
1869:42 D1 5C 80 40 20 20 20 A5
1871:F8 20 EA 4E EA 4D C5 6F EF
1879:C1 A7 78 5B F1 38 2C 7C 7E
1881:8A 9B 47 49 B8 DA E2 70 C2
1889:81 BB 0C B9 8C 9A 22 7B 15
1891:2A 80 A3 60 82 A0 38 5A D2
1899:0A 0A 7F F8 00 8D 4C 73
18A1:4B 2A A1 4D 09 02 82 48 A8
18A9:1C 82 8D 96 8C 90 4E 4E 35
18B1:AD 0F 20 2B 1C 0A 82 0D 4E
18B9:82 48 46 22 47 87 24 4C 15
18C1:47 C3 0C A0 A5 4D 4D 47 56
18C9:73 00 02 4C 4A 22 4B 4C 76
18D1:A8 12 4A 4B E8 4E 4B 85 75
18D9:21 43 FE F9 06 02 1B 82 DB
18E1:41 23 AA A1 C3 C3 60 0E E6
18E9:83 24 1D B5 62 AC 58 E3 3E
18F1:6A 16 2E F1 7C 0A 60 A1 30
18F9:E7 20 23 7B E4 68 FC 5A 5F
1901:EA 20 35 FC A4 36 3C C8 66
1909:E6 90 88 83 60 83 00 10 3D
1911:E8 1A 2C 41 E9 92 00 90 02
1919:28 1C 90 0C 91 E2 20 22 B3
1921:9E 03 02 31 C8 28 20 8A 68

```

```

1929:28 12 43 E6 20 2F 41 DA E5
1931:8D 8C 88 39 06 02 1A 88 E6
1939:E9 20 34 BC 81 20 46 A0 74
1941:41 42 43 1B 20 68 04 44 AD
1949:45 46 80 32 20 21 0A 3E BA
1951:FE A0 28 D5 A5 CC B1 93 E4
1959:72 E3 E4 A0 80 28 51 29 D4
1961:A0 90 8C 81 99 85 92 CD 87
1969:33 20 27 97 89 8E 93 41 8A
1971:1F B0 E3 A0 2E 94 92 81 50
1979:90 93 A0 8C 85 86 94 BA DF
1981:A0 B2 B5 E5 A0 EA 50 FD 15
1989:A0 00 00 68 F0 00 00 00 1A

```

Danny English is a frequent contributor who lives in Moreno Valley, California.

BASIC MOVE AND SAVE

By Daniel Lightner

Have you ever been in the middle of a great BASIC programming session when all of a sudden an *OUT OF MEMORY ERROR* message appears on the screen? Perhaps you've had a large program to halt in the middle of execution with a similar error message?

As a programmer, you may know that there's a 4K block of free RAM hidden under BASIC's ROM and RAM from 49152 to 53247. Wouldn't it be great if you could store some of your BASIC code there?

Well, you can with BAMOV and BASAV. These two utility programs for the 64 let you use this block of RAM that's usually reserved for machine language programs. They are particularly useful when you're using programs that require a lot of sprite or character data.

Getting Started

BASAV and BAMOV are written in machine language. To enter them, use MLX, our machine language entry program. See "Typing Aids" elsewhere in this section. When MLX prompts for starting and ending addresses for BASAV, respond with the following.

Starting address: C000
Ending address: C0C7

When entering BAMOV, respond with these addresses.

Starting address: CF62
Ending address: D001

Be sure to save each program before leaving MLX.

A Few Rules

Before these programs can be used, certain techniques must be employed and certain rules followed. Your large BASIC program must be divided into two parts. The second part of the program will be called by the first part during execution.

It's important to note that program 2 must be at least 42 bytes shorter than program 1. In most cases you won't have any problems determining this size differential, but here's a way to check. Load program 1 and type this line of code in direct mode.

```
PRINT INT(PEEK(46)*256)+PEEK(45)-2049
```

The value returned is the length of the program in bytes. Load the second program and enter the line again. To determine the difference, subtract the value given for program 2 from the value given for program 1. The number returned must be 42 or greater.

Special Coding

Program 1 must contain these or similar lines of code at the end of the program. Just be sure the line numbers are high enough to place the code at the end of the listing.

```

50000 GOSUB 50005
50001 SYS 53090: RETURN
50005 SYS 53090: GOTO10

```

When you want program 1 to call program 2, read its data, or whatever, have it GOSUB to line 50000. When program 2 has finished executing, the program will return normally to the next statement following the GOSUB 50000. However, it's not mandatory that control return to program 1.

Program 2 must also begin with whatever line the GOTO in line 50005 of program 1 dictates. In the above example, it's line 10. Remember to keep this number below 50000.

To pass control back to program 1, program 2 must end with a RETURN that is not part of any GOSUB routine in program 2.

A Demonstration

Two short demo programs labeled Prg1 and Prg2 are included to demonstrate how BASAV and BAMOV work.

These programs are written entirely in BASIC. To help avoid typing errors, enter them with The Automatic Proofreader. See "Typing Aids" again.

Running the Demos

Note that when Prg1 executes, it loads BAMOV and a file called Program2 into memory. Having the program load these two is not mandatory. You could load these two programs in direct mode before loading and running Prg1. If you decide to load them in immediate mode, delete lines 25 and 30 of Prg1. This will be better understood as we continue.

Load BASAV with the ,8,1 extension. Then type **NEW** and press Return. Now load Prg2 as you would any BASIC program. Before you go further, be sure there's a formatted disk in drive 8 in order to receive a relocated version of Prg2. Then type **SYS 49152** and press Return. The program will run, and the file will be saved as BAS-TMP. After the file has been saved, enter the following line of code in direct mode.

OPEN1,8,15,"R0:PROGRAM2=BAS-TMP":CLOSE1

It should be clear now that PROGRAM2 as listed in Prg1 is Prg2 relocated. Place a copy of BAMOV on the same disk as Program2. Reset the computer by either typing **NEW** or turning it off and on again. Load Prg1 and place the disk containing Program2 and BAMOV in drive 8. When you run the program, notice that control alternates between the two programs.

As its name implies, BAMOV is the BASIC mover. It pulls program 2 from beneath BASIC's ROM and places part of program 1 there. When activated again, it does the reverse.

When control is passed to line 50000 in program 1, it does a GOSUB to line 50005 so that when a RETURN is encountered, it will return to the next set of commands. At line 50005, BAMOV is activated, pulling program 2 into BASIC's memory while removing program 1. After it returns from the SYS call, the program encounters the GOTO10 command, and BASIC passes control to line 10 of program 2.

Program flow continues from there until it encounters a RETURN. At that

point, control returns to line 50001 following the GOSUB in line 50000 of program 1.

Note that this line must remain at the same location in memory. This is the reason for making sure that program 2 is at least 42 bytes shorter than program 1. Next, BAMOV is called again, and program 1 is put back in place. The RETURN in line 50001 returns control to the line that originally called the GOSUB50000, in this case line 65. All the switching back and forth may sound confusing, but it should become clear when you run the programs.

BASIC programs that require sprite and character data can read the data into memory and then pass control to the second program. But remember that this can only work as long as the second program is shorter than the first program.

BASAV

```
C000:AD 0E DC 29 FE 8D 0E DC 31
C008:A5 01 29 FE 85 01 AD 0E 4B
C010:DC 09 01 8D 0E DC A9 C1 34
C018:8D 18 03 A9 34 8D 14 03 64
C020:A5 2D 8D 00 A0 A5 2E 8D F6
C028:01 A0 A9 01 85 FB A9 08 0F
C030:85 FC A9 03 85 FD A9 A0 31
C038:85 FE A5 2D 8D B2 02 A5 A4
C040:2E 8D B3 02 A0 00 B1 FB 37
C048:91 FD 20 9F C0 A5 FB CD 72
C050:B2 02 D0 F2 A5 FC CD B3 65
C058:02 D0 EB A9 07 A2 BA A0 FF
C060:C0 20 BD FF A9 02 A2 08 A4
C068:A0 02 20 BA FF A6 FD A4 A5
C070:FE A9 00 85 FD A9 A0 85 91
C078:FE A9 FD 20 D8 FF AD 0E D5
C080:DC 29 FE 8D 0E DC A5 01 A3
C088:09 01 85 01 AD 0E DC 09 F7
C090:01 8D 0E DC A9 47 8D 18 23
C098:03 A9 31 8D 14 03 60 18 8A
C0A0:A5 FB 69 01 85 FB A5 FC 95
C0A8:69 00 85 FC 18 A5 FD 69 1C
C0B0:01 85 FD A5 FE 69 00 85 51
C0B8:FE 60 42 41 53 2D 54 4D 73
C0C0:50 00 00 00 00 00 00 6A
```

BAMOV

```
CF62:AD 0E DC 29 FE 8D 0E DC B1
CF6A:A5 01 29 FE 85 01 AD 0E CB
CF72:DC 09 01 8D 0E DC A9 C1 B4
CF7A:8D 18 03 A9 34 8D 14 03 E4
CF82:A9 01 85 FB A9 08 85 FC 1D
CF8A:A9 03 85 FD A9 A0 85 FE 2A
CF92:AD 00 A0 8D B2 02 AD 01 EF
CF9A:A0 8D B3 02 A0 00 B1 FB E8
CFA2:8D B4 02 B1 FD 91 FB AD 6D
CFAA:B4 02 91 FD 20 E0 CF A5 01
CFB2:FB CD B2 02 D0 E8 A5 FC AC
CFBA:CD B3 02 D0 E1 AD 0E DC 3A
CFC2:29 FE 8D 0E DC A5 01 09 D1
CFC A:01 85 01 AD 0E DC 09 01 3E
```

```
CFD2:8D 0E DC A9 47 8D 18 03 96
CFDA:A9 31 8D 14 03 60 18 A5 FD
CFE2:FB 69 01 85 FB A5 FC 69 2D
CFEA:00 85 FC 18 A5 FD 69 01 06
CFF2:85 FD A5 FE 69 00 85 FE CE
CFFA:60 00 00 00 00 00 00 CA
```

PRG1

```
EA 10 REM COPYRIGHT 1992
GJ 15 REM COMPUTE PUBLICATIONS
INTL LTD
GM 20 REM ALL RIGHTS RESERVED
HA 25 X=X+1:IFX=1THENLOAD"PROG
RAM2",8,1
AP 30 IFX=2THENLOAD"BAMOV",8,1
AJ 35 PRINT"{CLR}":POKE53280,0
:POKE53281,0
HA 40 PRINT"{2 DOWN}{2 RIGHT}
{7}THIS IS PROGRAM ONE O
F THE BAMOV DEMO."
MQ 45 PRINT"{DOWN}{2 RIGHT}PRO
GRAM TWO IS UNDER BASIC'
S ROM."
DQ 50 PRINT"{2 DOWN}{2 RIGHT}I
T WILL CLEAR THE SCREEN
{SPACE}AND"
CM 55 PRINT"{2 DOWN}{2 RIGHT}C
HANGE THE SCREEN AND BOR
DER COLORS"
ES 60 PRINT"{2 DOWN}{2 RIGHT}W
HILE DISPLAYING A MESSAG
E."
FB 65 FORT=1TO5000:NEXTT
AX 70 GOSUB50000
PX 75 POKE53280,0:POKE53281,0:
PRINT"{CLR}{2 DOWN}
{2 RIGHT}{7}BACK AT PROG
RAM ONE NOW!"
HE 80 END
RA 50000 GOSUB50005
RQ 50001 SYS53090:RETURN
MX 50005 SYS53090:GOTO10
```

PRG2

```
EA 10 REM COPYRIGHT 1992
GJ 15 REM COMPUTE PUBLICATIONS
INTL LTD
GM 20 REM ALL RIGHTS RESERVED
EQ 25 PRINT"{CLR}":POKE53280,6
:POKE53281,6
PX 30 PRINT"{2 DOWN}{2 RIGHT}
{WHT}THIS IS PROGRAM TWO
OF THE BAMOV DEMO."
MX 35 PRINT"{2 DOWN}{2 RIGHT}W
HEN THIS PROGRAM FINISHE
S, IT WILL"
HR 40 PRINT"{2 DOWN}{2 RIGHT}R
ETURN CONTROL TO LINE 50
001"
XC 45 PRINT"{2 DOWN}{2 RIGHT}O
F PROGRAM ONE."
PC 50 FORT=1TO5000:NEXTT
PD 55 RETURN
```

Daniel Lightner, a frequent contributor, lives in Sidney, Montana.

NOAH'S READER

By Daniel Lightner

Last year (July 1991) we published Noah's Arc, a program that creates self-dissolving archive (SDA) files. People who use that program will find this short utility program for the 64 valuable.

Archiving is a convenient method for combining a number of related files into one master file. This process is convenient for uploading and downloading programs and instructions to and from a BBS. Many files and programs can be stored within one large file. When the SDA file is loaded and run, it dissolves into the original individual programs and saves them to disk.

The problem with archive files is that unless you have the filenames written down, there isn't any way of knowing the contents of the archived file. This is especially true if you have just downloaded a new file from a BBS or have come across a forgotten SDA file in your library. Dissolving the file will do the trick, but it's time-consuming and a bit awkward.

Noah's Reader solves this problem. Noah's Reader reads the beginning of the SDA files from disk and lists the names of the files that are stored within the archive file.

Entering the Program

Noah's Reader is written in machine language and will have to be entered using MLX, COMPUTE's machine language entry program. See "Typing Aids" elsewhere in this section. When MLX prompts for starting and ending addresses, respond with these values.

Starting address: 0801

Ending address: 09F7

Make sure that you save a copy of Noah's Reader before you exit MLX.

Running the Program

Noah's Reader loads and runs like a BASIC program. The first thing it does is to ask for an SDA filename. It then searches drive 8 for that filename and reads information until it locates the various filenames.

Noah's Reader then lists those files to the screen. The listing can be stopped by pressing any key. When the key is released, the listing contin-

ues until it prints the names of all of the archived files.

Run Noah's Reader again to read another SDA file.

NOAH'S READER

```
0801:0B 08 0A 00 9E 32 30 36 2E
0809:31 00 00 00 A0 00 8C 20 EF
0811:D0 8C 21 D0 B9 8F 09 C9 C5
0819:FF F0 07 20 D2 FF C8 4C BC
0821:15 08 A0 00 20 FA 08 B9 88
0829:85 09 20 D2 FF C8 C0 0A 1E
0831:D0 F5 20 FA 08 A9 3E 20 5E
0839:D2 FF 20 15 09 AC 34 03 6E
0841:A2 00 BD 81 09 99 35 03 8E
0849:EE 34 03 C8 E8 E0 05 D0 70
0851:F1 CE 34 03 AD 34 03 A2 AB
0859:35 A0 03 20 BD FF A9 02 D1
0861:A2 08 A0 02 86 FC 20 BA 1C
0869:FF 20 C0 FF 20 CC FF A5 73
0871:BA 20 B4 FF A9 6F 85 B9 4D
0879:20 96 FF 20 A5 FF C9 30 32
0881:D0 0D 20 A5 FF C9 30 D0 F3
0889:06 20 AB FF 4C 96 08 20 07
0891:AB FF 4C E7 08 A2 02 20 6E
0899:C6 FF 20 FA 08 A9 01 85 2F
08A1:FB 20 F1 08 20 E4 FF 85 90
08A9:FE 20 07 09 A5 FB C9 B0 14
08B1:D0 F2 A5 FC C9 09 D0 EC 6C
08B9:20 E4 FF 85 FD A2 00 20 06
08C1:E4 FF C9 2C F0 08 20 D2 FA
08C9:FF E8 E4 FD D0 F1 A9 0D 3F
08D1:20 D2 FF 20 F1 08 20 F1 8A
08D9:08 20 E4 FF A5 CB C9 40 C2
08E1:D0 FA C6 FE D0 D2 A2 00 F8
08E9:20 C6 FF A9 02 4C C3 FF 1F
08F1:20 E4 FF 20 E4 FF 4C E4 F1
08F9:FF A9 0D 20 D2 FF 20 D2 C1
0901:FF A9 9A 4C D2 FF 18 A5 02
0909:FB 69 01 85 FB A5 FC 69 C5
0911:00 85 FC 60 A0 00 A9 00 82
0919:0D 34 03 20 E4 FF C9 00 1C
0921:F0 F9 C9 14 F0 41 C9 7B 40
0929:B0 F1 C9 11 F0 ED C9 13 40
0931:F0 E9 C9 1D F0 E5 C9 22 16
0939:F0 E1 C9 2C F0 DD C9 0D D7
0941:F0 10 AC 34 03 C0 14 F0 DC
0949:D2 20 D2 FF 20 5E 09 4C FF
0951:1C 09 AC 34 03 C0 00 F0 98
0959:C2 20 D2 FF 60 AC 34 03 50
0961:99 35 03 EE 34 03 60 AC F7
0969:34 03 C0 01 B0 03 4C 1C C4
0971:09 20 D2 FF 38 AD 34 03 4E
0979:E9 01 8D 34 03 4C 1C 09 40
0981:2C 50 2C 52 46 49 4C 45 9D
0989:4E 41 4D 45 20 3F 93 9A D0
0991:0D 20 20 20 20 20 20 4E 48
0999:4F 41 48 27 53 20 53 44 25
09A1:41 20 52 45 41 44 45 52 F2
09A9:0D 20 20 20 20 20 20 32 32
09B1:43 4F 50 59 52 49 47 48 67
09B9:54 20 31 39 39 32 0D 43 A7
09C1:4F 4D 50 55 54 45 20 50 76
09C9:55 42 4C 49 43 41 54 49 46
09D1:4F 4E 53 20 49 4E 54 4C 04
09D9:20 4C 54 44 0D 20 20 20 27
09E1:20 20 41 4C 4C 20 52 49 C9
09E9:47 48 54 53 20 52 45 53 99
09F1:45 52 56 45 44 0D FF 00 B0
```

Daniel Lightner is a frequent contributor who lives in Sidney, Montana.

LOCATE

By Farid Ahmad

Programmers who use BASIC are familiar with the various tricks for positioning text on a screen. Most use various PRINT statements and a lot of trial and error, but now there's a better way.

Locate is a short machine language routine for the 64 that provides BASIC programmers with two commands for cursor positioning and text color adjustment. Although the program is written in BASIC, it stores its machine language subroutine in a BASIC REM statement. This technique provides the speed of machine language with the convenience of BASIC.

Preparing Locate

Notice that Locate's first line contains a REM followed by 73 periods. It looks strange, but it's important not to change this line in any way. Since this line fills two screen lines, enter it without a space between the line number and the word REM. If you include the space, your cursor will drop down a line after you type the final period. Should that occur, cursor back up to the line and press Return.

Locate is written entirely in BASIC. To help avoid typing errors, use The Automatic Proofreader to enter the program. See "Typing Aids" elsewhere in this section. Be sure to save a copy of the program when you've finished.

Load and run the program. Now list it again. You'll see that Locate's first line number is missing and the line itself contains a number of meaningless characters. Next, delete lines 30-90. Delete a line by cursoring to an empty spot on the screen, typing 30, and then pressing Return. Do this for lines 30-90. Finally, the program will consist of only two lines: the unnumbered line 10, which contains the meaningless symbols, and line 20. Save this two-line program with the usual SAVE command.

Using the Program

Before starting to write a BASIC program, load this two-line program. Now start writing your program with a line

number greater than 20. When you want to position text, the following two commands are available.

SYS AT, row, column, color

The row may be from 0-24 and the column from 0-39. The color may be from 0-15, the usual Commodore colors. This parameter will effect the color of following text. Values outside these limits will produce an **ILLEGAL QUANTITY ERROR** message.

For example, **SYS AT, 5, 0, 1** will position the cursor at the beginning of the sixth screen line and change text color to white. The color parameter is optional. If you don't want to set the text color, omit this parameter and the preceding comma. **SYS AT, 5, 0** will position the cursor at the same place but will not change the text color. Spaces after the commas are also optional. Any **PRINT** statement that follows this or the following command will begin printing at the cursor position that you have indicated.

SYS CL, row, column, color

The syntax of this command is exactly the same as that of **SYS AT**, but it clears the screen before positioning the cursor. For example, **SYS CL, 0, 0, 1** will clear the screen, position the cursor at the upper left corner, and set text color to white. As with **SYS AT**, the color parameter is optional.

Other Considerations

The machine language routine in **Locate** is relocatable. It will work correctly even if the start of BASIC pointer has been changed. The only condition is that the two lines of **Locate** be the first two lines of the program. The line numbers, however, may be changed with a renumbering utility.

The variables **AT** and **CL** are defined by **Locate** as the starting addresses of the **Locate** routines. These variables must not be used elsewhere in the program, or the program might crash.

If you want to use **Locate** with an existing program, you'll need a merge utility, such as the **MERGE** command in **METABASIC**. Renumber your program so that the first line number is greater than 20. Then merge it with **Locate**.

A Demonstration

Demo is a demonstration program that illustrates some of the ways **Locate** commands can be used and modified. It's also written in BASIC and should be entered with **Proofreader**.

With a merge program, you can combine the two programs later. If you don't have a merge program, load and run **Proofreader**, load the two-line **Locate** program, and then enter **Demo**, starting with line 30.

The Technique

The technique used with **Locate** is a convenient way of adding short machine language routines to BASIC programs. A few things must be kept in mind, however. First, the **ML** routine must not contain the number 0. This is because 0 is reserved by the BASIC interpreter to mark the end of a BASIC line. Since 0 is the **ML** instruction for **BRK**, it's seldom required. It may be needed, however, as the argument of an **ML** command. It's usually possible to get around this problem. For example, to load the **X** register with 0, use **LDX#1, :DEX**.

Note the quotation mark at the beginning of the first line. If this is not included, the **ML** numbers will be interpreted as BASIC tokens. This will still work, but the resulting list may look a bit strange. The quote itself may also produce some problems. Once the quote is encountered, some of the graphic characters might be interpreted as control characters. When the program is listed, the list may change colors, or the screen may be cleared. This is irritating, but it doesn't do any harm to the program. The best way to avoid this problem is to list the program from the second (or higher) line. Whether or not the quote is used, once the **ML** is in the **REM** statement, do not reenter the line by pressing **Return** over it. This will enter the line incorrectly and garble the **ML**. If the quote has been used, the line may look the same after reentering, but the damage may still have been done. This is because many graphic symbols have more than one **POKE** code, and the BASIC editor always stores the lower value in memory. So if your **ML** contains the instruction **JSR \$AEFD**, reentering the line will change this to **JSR\$ AEBD**, as **\$FD**

and **\$BD** are the **POKE** codes for the same graphic symbol.

Locate prevents this from happening by including enough delete characters in the line to delete the line number. Thus, the line cannot be reentered by mistake.

LOCATE

```
EQ 10 REM".....
.....
.....
.....
EC 20 CL=PEEK(43)+256*PEEK(44)
+14:AT=CL+5
KD 30 DATA20,20,20,20,20,20,20,20
KG 40 DATA{2 SPACES}169,147,03
2,210,255,032,253,174,03
2,158,183,134,002,032,25
3,174,032
AE 50 DATA{2 SPACES}158,183,13
8,168,166,002,224,025,17
6,033,192,040,176,029,02
4,032,240
CJ 60 DATA{2 SPACES}255,160,00
1,136,177,122,170,224,04
4,208,014,234,032,253,17
4,032,158
BD 70 DATA{2 SPACES}183,224,01
6,176,004,142,134,002,09
6,162,014,076,139,227
RH 80 FORI=0TO72:READA:CK=CK+A
:POKE CL-8+I,A:NEXT
DS 90 IFCK<8427 THENPRINT"ERR
OR IN DATA STATEMENTS":E
ND
```

DEMO

```
MQ 30 A$="L O C A T E":B$="LOC
ATE"
AP 40 PP=15
MA 50 POKE53280,0:POKE53281,0
KS 60 SYSCL,10,09,1:PRINTA$
AB 70 FORA=1 TO 09
DR 80 SYSAT,A,A,A:PRINTB$
BD 90 SYSAT,A,35-A,A:PRINTB$
MH 100 NEXT
HA 110 FOR A=13 TO 1 STEP -1
GP 120 SYSAT,A+10,A+10,15-A:PR
INTB$
SK 130 NEXT
QM 140 FOR A=1 TO100
QC 150 SYSAT,10,09,A-INT(A/15)
*15:PRINTA$
CP 160 NEXT
SQ 170 SYSCL,5,3,1
BJ 180 PRINT"LOCATE ALLOWS YOU
TO POSITION TEXT"
FE 190 SYSAT,7,5,2
DR 200 PRINT"ANYWHERE"
AX 210 SYSAT,9,13
MK 220 PRINT"ON"
QG 230 SYSAT,11,15
QC 240 PRINT"THE
BH 250 SYSAT,13,20
CJ 260 PRINT"SCREEN"
```


PROGRAMS

```

XX 270 SYSAT,PP,5,3
SJ 280 PRINT"IN"
FR 290 SYSAT,PP,10,6
FJ 300 PRINT"ANY"
FB 310 SYSAT,PP,15,11
PR 320 PRINT"COLOR"
MM 330 FORA=0 TO 15:SYSAT,PP,2
    2+A,A
QQ 340 PRINT"!"
BH 350 NEXT
    
```

Farid Ahmad is a frequent Gazette contributor. He lives in Islamabad, Pakistan.

BUG-SWATTER

A portion of the machine language listing for Blanker in the August 1992 issue was omitted. We regret the inconvenience it may have caused some readers. Here is the entire listing.

If you have already entered and saved the earlier portion, load and run MLX, responding with the following starting and ending addresses.

Starting address: 0247

Ending address: 0763

Now select Load File from the MLX menu and load the saved file. Then begin entering data from address 03D7.

After you have saved the entire program, remember that it must be converted to GEOS format with the converter program in the August issue.

BLANKER

```

0247:0F 03 15 BF FF FF FF 80 B2
024F:00 01 BF FF FD B0 00 0D 4B
0257:A0 00 05 A0 00 05 A0 00 AB
025F:05 A0 00 05 A0 00 05 A0 0E
0267:00 05 A0 00 05 A0 00 05 70
026F:A0 00 05 A0 00 05 B0 00 E3
0277:0D BF FF FD 80 00 01 FF D7
027F:FF FF 4F FE 72 20 00 04 75
0287:3F FF FC 83 05 00 00 04 2F
028F:BA 2C 00 04 53 63 72 6E B7
0297:20 42 6C 61 6E 6B 65 72 3E
029F:56 31 2E 30 00 00 00 00 E3
02A7:43 68 61 72 6C 65 73 20 BA
02AF:57 2E 20 42 6F 7A 61 72 AD
02B7:74 68 20 0F 2C 00 00 44 AA
02BF:65 73 6B 20 61 63 63 65 87
02C7:73 73 6F 72 79 20 66 6F FF
02CF:72 20 62 6C 61 6E 6B 69 2D
02D7:6E 67 20 74 68 65 20 47 98
02DF:45 4F 53 20 73 63 72 65 3A
02E7:65 6E 2E 0F 34 00 00 20 B2
02EF:4E C1 20 B7 C1 00 00 99 73
02F7:08 40 1F 20 53 C2 00 C8 64
02FF:00 00 40 01 A9 80 85 2F A5
0307:20 B7 C1 1F 85 1E 29 9C C8
030F:03 20 B6 06 20 31 07 20 C9
    
```

```

0317:E6 06 A5 02 C9 02 F0 38 57
031F:A9 00 85 39 A9 04 8D A2 59
0327:84 A9 69 8D A1 84 A9 04 56
032F:8D A4 84 A9 69 8D A3 84 9D
0337:A9 01 85 3B A9 3F 85 3A 46
033F:A9 C7 85 3C A5 16 0A A8 C2
0347:88 88 B9 29 07 8D 9B 84 A7
034F:C8 B9 29 07 8D 9C 84 60 06
0357:AD 11 D0 09 10 8D 11 D0 CC
035F:A9 30 85 01 20 B7 C1 1E 88
0367:29 1F 85 9C 03 20 A5 C1 E9
036F:00 C8 00 00 40 01 20 B7 A5
0377:C1 99 08 00 60 40 1F 4C 54
037F:3E C2 A9 00 8D 9C 84 A9 1C
0387:00 8D 9B 84 A9 35 85 01 DA
038F:AD 11 D0 29 EF 8D 11 D0 06
0397:60 A9 40 8D 9C 84 A9 C7 A3
039F:8D 9B 84 A9 A0 85 03 A9 49
03A7:00 85 02 A9 00 85 05 85 8F
03AF:06 A9 5A 85 07 60 A6 05 D2
03B7:A4 06 B9 28 05 A8 B1 02 7B
03BF:3D 20 05 91 02 20 0A 05 CF
03C7:18 A9 0A 65 02 85 02 90 96
03CF:02 E6 03 A5 03 C9 BF D0 DA
03D7:04 A5 02 C9 40 D0 D7 A9 C4
03DF:A0 85 03 A9 00 85 02 20 CC
03E7:0A 05 C6 07 D0 0A A9 04 83
03EF:8D 9C 84 A9 93 8D 9B 84 9D
03F7:60 E6 06 A5 06 C9 0B D0 41
03FF:04 A9 00 85 06 E8 E0 08 68
0407:D0 02 A2 00 86 05 00 7F D4
040F:FB DF FE EF BF FD F7 03 D4
0417:07 01 09 04 06 00 08 05 89
041F:02 00 A9 05 8D 9C 84 A9 3F
0427:48 8D 9B 84 A0 09 A9 FF EE
042F:99 8E 08 88 D0 FA 60 20 84
0437:76 05 90 28 18 20 33 C1 B9
043F:E6 18 A5 18 C9 C8 F0 14 5E
0447:A9 00 38 20 33 C1 20 87 95
044F:C1 AD 0A 85 C9 28 B0 04 91
0457:A9 C8 85 18 20 96 05 A9 A7
045F:10 8D 4E 06 06 AC 50 06 59
0467:B9 8E 08 C9 FF F0 12 85 FA
046F:18 98 0A A8 B9 7A 08 85 C2
0477:08 C8 B9 7A 08 85 09 38 35
047F:60 20 C6 05 60 AC 50 06 45
0487:A5 18 C9 C8 F0 21 99 8E FB
048F:08 98 0A A8 A5 08 99 7A 88
0497:08 C8 A5 09 99 7A 08 EE D0
049F:50 06 AD 50 06 C9 0A D0 48
04A7:05 A9 00 8D 50 06 60 A9 7A
04AF:FF 99 8E 08 60 CE 4E 06 51
04B7:D0 19 AD 4F 06 18 69 08 84
04BF:8D 4F 06 C9 60 90 0A A9 C2
04C7:04 8D 9C 84 A9 AA 8D 9B BF
04CF:84 18 60 AD 4F 06 8D 98 4D
04D7:08 20 87 C1 AD 0B 85 C9 67
04DF:FA B0 D2 85 19 AD 0A 85 5D
04E7:85 18 A9 01 85 05 A9 40 D1
04EF:85 04 A2 18 A0 04 20 69 50
04F7:C1 A5 13 85 09 A5 12 85 8D
04FF:08 20 3F C1 90 01 60 CE 30
0507:98 08 F0 A9 20 87 C1 30 EA
050F:1D E6 08 D0 02 E6 09 A5 D2
0517:09 C9 01 D0 04 A5 08 C9 D5
051F:40 D0 DE A9 00 85 09 A9 C5
0527:00 85 08 4C 12 06 E6 18 E6
052F:A5 18 C9 C8 D0 CB A9 00 E0
0537:85 18 4C 12 06 10 07 00 33
053F:A9 06 8D 9C 84 A9 60 8D 34
    
```

```

0547:9B 84 A9 01 85 02 60 A9 24
054F:C8 85 06 A2 00 86 03 A9 D3
0557:00 85 05 20 3C C1 A2 28 BB
055F:18 A0 00 B1 0C F0 02 E6 C7
0567:05 6A 91 0C 08 18 A9 08 7D
056F:65 0C 85 0C 90 02 E6 0D 08
0577:28 CA D0 E7 A5 05 D0 11 D4
057F:C6 06 A5 06 D0 0B A9 04 8D
0587:8D 9C 84 A9 93 8D 9B 84 39
058F:60 A6 03 E8 E4 02 D0 BD F0
0597:E6 02 A5 02 C9 C8 90 04 01
059F:A9 C8 85 02 60 A9 27 85 FE
05A7:05 A9 D9 85 04 A9 8C 85 97
05AF:07 A9 A7 85 06 A2 8D A0 6A
05B7:19 B1 06 20 11 07 AD 27 A4
05BF:8C 91 06 88 D0 F3 18 A9 ED
05C7:28 65 06 85 06 90 02 E6 B5
05CF:07 CA D0 E3 60 A9 27 85 E5
05D7:05 A9 D9 85 04 A9 8C 85 C7
05DF:07 A9 A7 85 06 A2 0D A0 9A
05E7:19 20 1A 07 91 06 88 D0 C0
05EF:F8 18 A9 28 65 06 85 06 88
05FF:90 02 E6 07 CA D0 E8 60 E3
05FF:84 08 A0 00 91 04 4C 20 B7
0607:07 84 08 A0 00 B1 04 E6 78
060F:04 D0 02 E6 05 A4 08 60 2B
0617:93 04 AA 04 33 05 51 06 D9
061F:A9 00 85 3B A9 E0 85 3A 7A
0627:A9 5D 85 3C A9 07 85 03 4B
062F:A9 C9 85 02 20 56 C2 60 93
0637:A9 00 8D B5 84 A9 07 8D 8B
063F:9C 84 A9 59 8D 9B 84 60 C9
0647:A9 00 8D 9C 84 A9 00 8D FB
064F:9B 84 A9 01 85 16 20 9E F2
0657:07 60 A9 01 85 17 A5 16 2E
065F:C9 01 D0 22 60 A9 02 85 FF
0667:17 A5 16 C9 02 D0 17 60 A9
066F:A9 03 85 17 A5 16 C9 03 4F
0677:D0 0C 60 A9 04 85 17 A5 9F
067F:16 C9 04 D0 01 60 20 9E FE
0687:07 A5 17 85 16 A9 00 85 98
068F:09 A9 59 85 08 A9 00 85 7A
0697:0B A9 61 85 0A A9 2A 85 E8
069F:06 A6 16 A5 06 18 69 11 E9
06A7:85 06 CA D0 F6 A5 06 18 D0
06AF:69 06 85 07 20 2A C1 60 A0
06B7:81 13 49 07 01 11 37 02 9F
06BF:11 48 0B 10 10 38 08 0B 45
06C7:30 21 59 08 0B 30 32 61 BE
06CF:08 0B 30 43 6C 08 0B 30 A6
06D7:54 73 08 12 03 1A 01 08 97
06DF:12 03 2B 09 08 12 03 3C 76
06E7:11 08 12 03 4D 19 08 00 CF
06EF:21 08 00 00 02 09 6B 07 A0
06F7:21 08 00 00 02 09 76 07 BE
06FF:21 08 00 00 02 09 81 07 DC
0707:21 08 00 00 02 09 8C 07 FB
070F:92 FF E0 80 20 80 20 80 4E
0717:20 80 20 80 20 80 20 80 25
071F:20 FF E0 0E 00 04 BF 18 E1
0727:50 6C 65 61 73 65 20 53 FF
072F:65 6C 65 63 74 20 42 6C 03
0737:61 6E 6B 69 6E 67 20 4F 36
073F:70 74 69 6F 6E 3A 1B 00 59
0747:18 42 6C 61 6E 6B 1B 00 EC
074F:18 44 69 73 73 6F 6C 76 87
0757:65 1B 00 18 44 72 69 70 8F
075F:1B 00 18 54 69 6C 74 1B 44
0767:00 00 00 00 00 00 00 75
    
```